

Editorial

THIS ISSUE of the Polytechnic Open Library International Bulletin of Information Technology and Science (POLIBITS) includes ten papers by authors from six different countries: France, Germany, Ireland, Mexico, Spain, and USA. The papers included in this issue are devoted to such topics as machine translation, natural language processing, information retrieval, image processing, and classification algorithms for sensor applications in the context of the Internet of Things.

Christopher G. Harris from **USA** in his paper “A Framework to Build Quality into Non-Expert Translations” proposes a framework for obtaining high-quality translation from non-expert crowdworkers, by incorporating intermediate mechanisms such as ranking and editing in addition to translation. He demonstrates the effectiveness of the proposed framework in terms of translation quality, time, and cost. The proposed framework is robust against spammers, verifiable at different steps, and consistent with little variance in quality. The proposed framework is most appropriate when it is necessary to maximize translation quality and minimize cost.

César Marrades Cortés et al. from **Ireland** in their paper “Sports Video Anonymisation and Accuracy Prediction” suggests methods for preserving anonymity of persons in video recordings such as recording of school sport examinations. For the task, the authors use machine learning predictive models to investigate the factors that affect the anonymisation program’s performance on sports videos. On one hundred video inputs, they achieve an accuracy of 94% and a specificity of 95.2%.

Angélica Hernández Rayas et al. from **Mexico** in their paper “Marching Cubes Algorithm for Transforming Images” presents a modification of the Marching Cubes algorithm to make a three-dimensional representation from surface image discretized on volumetric pixels. For this, they connect each component continuously regardless of the values of the image surface. The modification aims to achieve this effect optimizing computational resources. The obtained results show high performance of the proposed modification.

Yuming Zhai et al. from **France** in their paper “Towards Recognizing Phrase Translation Processes: Experiments on English-French” study the processes that human translators, consciously or not, resort to in their translation activity apart from the literal translation. Such processes can be idiom equivalence, generalization, particularization, semantic modulation, etc. Several typologies exist to characterize such translation processes. The authors automatically classify these fine-grained translation processes. Their results show that they can distinguish non-literal translation from literal translation

with an accuracy of 87.09%, and 55.20% for classifying among five non-literal translation processes. This demonstrates that it is possible to classify automatically translation processes even with a small amount of annotated examples.

David Muñoz et al. from **Ireland** and **Mexico** in their paper “Named Entity Recognition Based on a Graph Structure” propose a graph structure for storage and enrichment of named entities that makes use of synonyms and domain-specific ontologies in the area of computing. They experimentally measure the performance of the proposed structure is measured and compared it with other NER classifiers. This work is useful for various areas of natural language processing, such as text mining and information retrieval.

Mónica Villaverde et al. from **Spain** in their paper “A Comparison of Adaptive and Non-Adaptive Ensemble Methods for Classification Systems” analyze work four different approaches to aggregation the estimations given by the ensemble of sensors in order to obtain the final classification. A number of classifiers are analyzed: ANN, plurality majority, basic weighted majority and stochastic weighted majority. They compare the majority-voting algorithm and the artificial neural network against two proposed adaptive algorithms based on weighted majority and without previous training. In this comparison, the authors not only take into account the accuracy of each algorithm but also adaptation. The experiments show that the artificial neural network is the most accurate proposal, whereas the most innovate proposed stochastic weighted voting is the most adaptive one.

Olga Kolesnikova and **Alexander Gelbukh** from **Mexico** in their paper “Dictionary and Corpus-Based Study of Lexical Functions in Spanish” study semantic and contextual characteristics of four types of verb-noun collocations in Spanish, corresponding to a different so-called lexical function. They explain how the typology of lexical functions can be viewed as a consistent way to classify collocations by their semantic and syntactic patterns. They examine how different lexical functions as well as free word combinations can be identified automatically by supervised machine learning methods. The authors show that contextual characteristics are not powerful enough to discriminate among subtle semantic differences between lexical functions.

Sannikumar Patel et al. from **Ireland** in their paper “Word Embeddings and Length Normalization for Document Ranking” address the problem of the document length bias in information retrieval techniques based on distributed word

representations. They argue for that document length normalization is useful to address the length bias problem while using embedding-based ranking. They present experimental evidence for their claim. The proposed method of length normalization significantly improves the Mean Average Precision, by up to 47% over a baseline that uses a simple embeddings-based algorithm.

Rafael Gallardo García et al. from **Mexico** in their paper “Facial Recognition using Convolutional Neural Networks and Supervised Few-Shot Learning” present a feature-based face recognition method, consisting of two separated processes. First, a pre-trained CNN-based face detector extracts the locations and features of the faces found in the input images. Then the output of this pre-trained face detector is used to train the models for the classifiers that are used to find unknown faces in new images. The authors show that their method achieves high accuracy on datasets with several individuals but few training samples.

Björn Buchhold and **Jörg Dallmeyer** from **Germany** in their paper “Zero-Shot Learning for Topic Detection in News Articles” present a method to detect topics in news articles and represent them by a descriptive document. The authors use a

neural network that operates on two kinds of inputs: the full texts of the descriptive documents passed through the same recurrent encoder network, and a proprietary NLP pipeline and knowledge base used to recognize named entities and significant keywords. The authors evaluate and compare several model configurations on two datasets, a large one automatically created from Wikipedia and a smaller one created manually.

This issue of the journal will be useful to researchers, students, and practitioners working in the corresponding areas, as well as to public in general interested in advances in computer science, artificial intelligence, and computer engineering.

Alexander Gelbukh,
Oscar Camacho
Instituto Politécnico Nacional,
Mexico City, Mexico
Editors-in-Chief

A Framework to Build Quality into Non-Expert Translations

Christopher G. Harris

Abstract—When performed correctly, crowdsourcing can produce near-expert quality translations both quickly and inexpensively; however, the quality obtained from crowdworkers is rarely consistent. We propose a framework to obtain high-quality work from non-expert crowdworkers by incorporating intermediate mechanisms such as ranking and editing in addition to translation. We conduct three empirical experiments in which we explore the impact of these framework mechanisms on translation quality, time, and cost. We also demonstrate that our proposed framework is robust against spammers, verifiable at different steps, and consistent with little variance in quality. Our framework achieved a higher BLEU score than professional translators at a fourteenth of the cost but required about 50 percent more time to complete. Therefore, it is most appropriate when a task requester wishes to maximize translation quality and minimize cost.

Index terms—Translation quality, crowdsourcing, translation framework.

1 INTRODUCTION

MACHINE TRANSLATION (MT) tools like Google Translate have recently made impressive strides in quality by implementing deep learning techniques. However, MT tools are unable to match the nuanced advantages that professional translations can provide, such as avoiding mistranslations, applying political and social correctness to translated text, and for many free online tools, maintaining confidentiality. One issue with professional translations is that the costs can be excessive, particularly with low-resource languages. To translate a corpus from Tamil to English, German (2001) calculated the cost of each translated word at \$0.36. However, a more recent estimate of the cost of translation for more common languages such as Spanish, French, and Chinese (obtained from websites proz.com, slator.com, and wordminds.com) range from \$0.27 to \$0.33 per word. These resources demonstrate that professional translation tasks between common languages can quickly become prohibitively expensive, even for moderate-sized translation efforts. Automatic evaluation metrics such as BLEU (Papineni et al., 2002) have been shown to correlate well

with human evaluations for machine translations (e.g., Hobson et al., 2007, Wu et al., 2016). Unfortunately, payments made for non-expert translations are often weakly correlated with the resulting output quality.

The growth of the internet has led to a noteworthy increase of choices for translations – a substantial number of non-experts, especially crowdworkers, were now available to perform the same tasks previously relegated to experts. Crowdsourcing platforms such as Amazon Mechanical Turk (AMT) have become prominent marketplaces for translations, particularly because they offer easy matchmaking services between task requester and worker. These platforms promote the benefits of having many tasks and workers available at any one time, offering inexpensive labor to requesters (and new opportunities to earn money for workers). These platforms focus on smaller, self-contained tasks with simple instructions (called microtasks) that can be performed quickly. It is often up to the requesters to provide checks on quality, which can be challenging when translating from a language the requester knows to one that they may not understand (or vice versa).

An ongoing concern regarding the use of crowdsourcing platforms such as AMT is that the quality of outputs can vary considerably. Moreover, there is an inherent misalignment between task requester and crowdworker; while both task requesters and workers desire the task to be accomplished quickly, most task requesters seek to maximize quality and minimize cost. However, since crowdworkers are perceived to be anonymous and transient, a sizeable subset of them, called spammers, seek to maximize their earnings with little concern for the output quality, giving rise to a market of imperfect information (Akelof, 1978). From a requester’s perspective, mechanisms to maximize quality and speed while minimizing cost need to be designed into any robust text translation system. Few requesters know how to do this or understand which mechanisms will provide the largest improvement in translation quality.

Over the last decade, researchers have tried a variety of approaches to maximize output quality while simultaneously minimizing cost. Although each researcher has independently demonstrated techniques to provide near-expert translation quality, there has been a scant focus on providing a comprehensive framework or set of tools to enhance these efforts. Moreover, none of these approaches examined the optimal situation in which to employ each mechanism. In this

Manuscript received on May 15, 2019, accepted for publication on July 23, 2019, published on December 30, 2019.

Christopher G. Harris is with Dept. of Computer Science, University of Northern Colorado, Greeley, CO 80639, USA (e-mail: christopher.harris@unco.edu).

paper, we seek to provide these, as well as to answer the following research questions.

1. Which components of our proposed framework can provide the greatest increase in translation quality with respect to time and cost?
2. How does adding additional crowdworkers in each step of the framework affect quality, time and cost?

Our paper makes the following contributions. We define distinct components of crowd-based translation tasks and examine how the flow of these components in a framework can affect quality. We examine these framework components in turn and discuss whether the use of each is most appropriate based on time, cost, and improvement in quality. We conduct experiments using non-experts (crowdworkers) that empirically examine each metric in our framework. We discuss our findings with respect to the overall framework and crowdsourcing in general.

2 BACKGROUND AND RELATED WORK

For over a decade, researchers have shown that non-experts hired from crowdsourcing platforms have been able to translate text with quality that approaches those produced by experts. Kittur et al. (2008) were one of the first to compare crowdworkers to experts in an NLP task. Workers were asked to rate Wikipedia articles according to several factors, such as the quality of writing, accuracy, and structure, on a seven-point Likert scale. Initially crowdworker ratings were poorly correlated with those made by experts ($r=0.5$); however, by redesigning their experiment to improve quality, the correlation with the expert ratings improved in a subsequent experiment ($r=0.66$). They suggested designing tasks to encourage accurate responses over malicious or random ones (i.e., spam), but they did not discuss how task flow can improve quality.

Snow et al. (2008) used AMT for non-expert annotations for five NLP tasks. Their majority voting approach only required a few non-expert labels to achieve near-expert quality. Callison-Burch (2009) used AMT to evaluate MT outputs using a few targeted strategies. In one approach, they hired non-experts to create reference translations in several languages, reinforcing the ease of obtaining near-expert quality translations quickly and inexpensively.

Cost savings using crowdsourcing platforms can be substantial, even when redundant quality checks are designed into the system. Hoffmann (2009) indicates that using crowdworkers for translating transcription services can save a company 33% compared with using in-house staff. Harris and Xu (2011) found using non-expert translators for translated transcriptions from Chinese to three other languages were, on average, 1/23rd the cost of professional translators, but translation quality was comparable. Novotney and Callison-Burch (2010) found professional translation costs to be thirty times as expensive as using crowdworkers.

The relationship between compensation and quality is often confounding. For example, Gillick and Liu (2010) experimented with different amounts of compensation for a text summarization exercise and found that a lower compensation (\$0.07) resulted in better quality. They surmised that a lower amount attracted crowdworkers who prioritized quality over making money, although the pool size of crowdworkers with such a priority on quality was small. However, in another study, Aker et al. (2012), found that higher payments resulted in better quality for objective tasks that contained verifiable answers (e.g., performing a calculation). In our study, we examine the relationship between quality and compensation provided to non-expert translators.

Using crowdworkers to do portions of a task has been examined previously, but only as inputs to MT algorithms. Buzek et al. (2010) used AMT to create paraphrase lattices as MT inputs. Two tasks were established: one to create the paraphrase lattices, and another one to verify the generated paraphrases. They found that if paraphrasing by crowdworkers targeted the most challenging areas of the lattice, TER score of the resulting translation could be improved.

A few researchers have explored the use of the crowd for different subtasks of a single translation project. Bloodgood and Callison Burch (2010) produced test sets for MT systems using crowdworkers instead of professionals and found that the quality of these crowd-created inputs matched those made by professionals. Zaidan and Callison-Burch (2009) examined the benefits of redundancy in translating, editing, and voting. They concluded that redundancy provides tangible benefits, but they do not indicate the relative benefits. Yan et al. 2016 examined creating a two-stage model with translator-editor pairs on AMT. They found that randomly pairing translators and editors provided the best quality. Hourcade and Gehrt, (2015) used crowdworkers in a two-step process: first to summarize medication warnings and then vote on the best summarization. El-Haj et al. (2017) obtained semantic labels for 250 words in six languages finding that a two-stage filtering process they used with crowdworkers improved quality and reduced spam. We build upon this notion in this paper.

The common limitation in these previous studies is that they focus on the use of crowdworkers only with specific components of translation and text summarization tasks – and do so convincingly – but do not provide an overall framework or guidance to follow. Our goal is to derive a framework that can be followed to build quality into translations from start to finish.

3 CROWDSOURCING FRAMEWORK

Translations frequently involve multi-lingual and bi-lingual workers, each of whom translates a snippet of text in a source language into snippets of text in (one or more) target languages. Text summarization, on the other hand, only requires monolingual non-experts, thus appealing to a larger pool of crowdworkers. MT tools such as Google Translate are

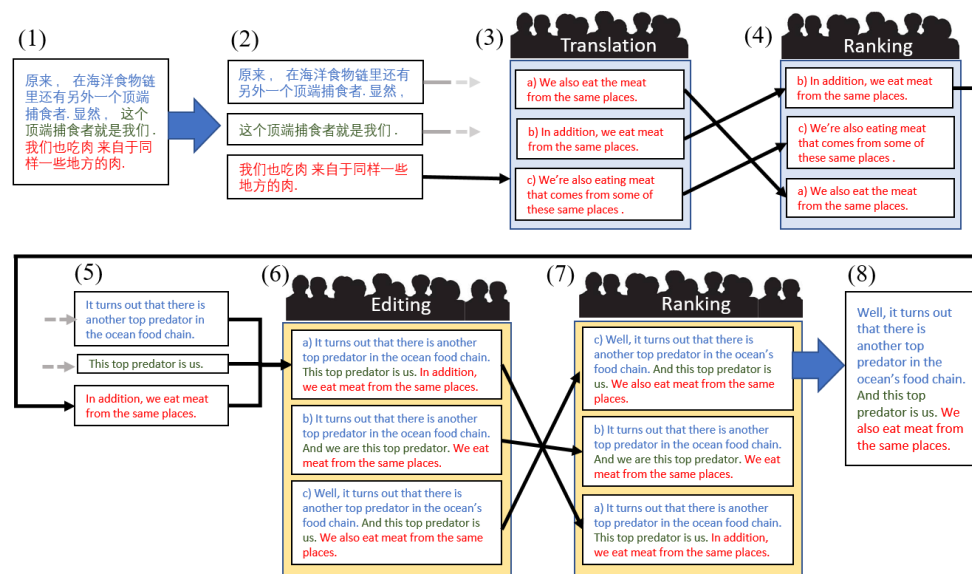


Fig. 1. An example of the 8 steps of the crowdsourcing framework given in Figure 1.

improving, but still lack the nuances of a human translator; to this end, we focus on providing translations with a focus on quality. Since crowdsourcing platforms are designed to provide microtasks or simple tasks each requiring a few minutes at a time, we also focus on keeping each task as atomic as possible with simple instructions. We seek to develop a framework with the following qualities:

- **Robust:** Our framework should be impervious to low-quality inputs from a malicious crowdworker or spammer.
- **Verifiable:** We should be able to evaluate our metrics after each crowdworker-dependent step in our framework.
- **Consistent:** To the extent possible, the task should be deterministic; the same inputs should produce approximately the same outputs, even with a different set of crowdworkers.

A Framework Components

Five crowdsourcing-dependent components are used in our proposed framework. To keep the pool of potential crowdworkers as large as possible for translations (which, due to the economics of supply and demand, lowers the cost and reduces the task completion time), we want as few components as possible to rely exclusively on multi- and bi-lingual crowdworkers

- **Ranking:** This component asks crowdworkers to rank text in order of relative preference. Ranking is helpful in situations where users have few choices and can discriminate between the choices; if there are many choices to choose from, scoring each item (on a Likert scale) may make more sense than ranking them. Studies using crowd ranking, such as that by Goto (2015) have demonstrated their effectiveness. Ranking does not depend on multi- or bi-lingual crowdworkers.

- **Translation:** Considered the core component, this is the task in which translated versions of the input text are generated.
- **Editing:** When a translation is achieved by using different crowdworkers, the overall tone and flow are affected. Editing “smooths” the document by improving the flow between segments of text, removing redundancies, and making the tone of the article consistent. Editing does not depend on multi- or bi-lingual crowdworkers.
- **Disassembly:** Divide a document (or collection of documents) into segments (subsets of the document of approximately consistent size). Disassembly is usually accomplished through automation.
- **Reassembly:** Recombine the translated segments into a single document. Like disassembly, reassembly is usually accomplished through automation.

B Framework Flow

We begin with the document in step 1 of Figure 1. For discussion, we consider it a single document although it could be comprised of a set of documents. In step 2, the document is automatically broken into n segments (t_1, t_2, \dots, t_n) – each segment can be as small as a sentence or as large as a paragraph. In step 3, each segment is submitted separately for translation by m non-experts. We note that for smaller segment sizes (i.e., a sentence), the lack of context may make translations challenging; therefore, we provide the entire document to each translator. This results in m translations for each segment, (e.g., $t_{n(1)}, t_{n(2)}, \dots, t_{n(m)}$ would be the translated segments for t_n).

Next, for each of the n segments, the m translated segments are then ranked by a separate pool of crowdworkers (step 4). The use of a separate pool of crowdworkers helps insure that

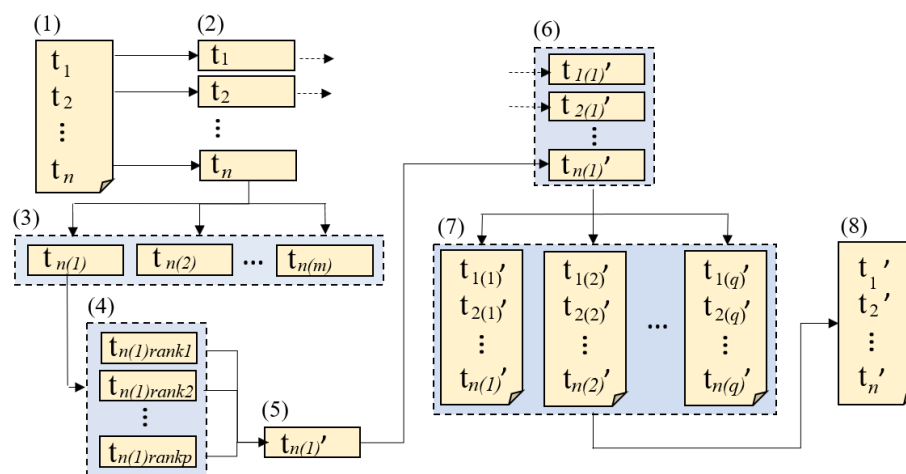


Fig. 2. Our framework for conducting translations and text summarizations using non-experts. Key tasks include translations (step 3), ranking translations (step 4), reassembly of the top-ranked translations for each segment (step 6) and a ranking of edits of the reassembled segments (step 7).

the output of a malicious crowdworker or spammer does not propagate past this stage. It is unique to our framework.

Once the m translations are ranked $1 \dots m$, the highest-ranked segment for each of the n segments (denoted as $t_{1(1)}' \dots t_{n(1)}'$ or simply $t_1' \dots t_n'$) is retained. These are then automatically reassembled in the original order (step 5). Although at this point we have a translation of the full document, it is really a rough combination of translations of different segments of text and is unlikely to flow together well. To accomplish this, we edit the translated document to ensure the translation maintains the correct context and follows a consistent tone. We accomplish this by having the document edited by q crowdworkers (step 6), and these outputs are ranked by another pool of crowdworkers (step 7) to obtain the final document (step 8). This final document is evaluated for quality using BLEU. Figure 2 provides an example showing a Chinese document divided into segments of a single sentence each.

4 EXPERIMENT DESIGN

With many variables to consider, we wish to optimize our framework. We conduct experiments to determine the payment amount, the number of crowdworkers, and the segment size necessary to produce expert-quality translations.

A. Data

We use a publicly-available dataset: the TED 2013 parallel corpora from (Tiedemann, 2012). We randomly select three transcripts, containing an average of 124 sentence pairs and 2299 English words each. For these transcripts, we obtain the parallel corpus for Chinese-English and French-English. As in Callison-Burch (2009), our objective was to reduce the use of available MT tools by crowdworkers, which would be viewed as cheating; we provided images of each text segment to prevent workers from copying and pasting text into an MT tool.

B. Metrics

We use BLEU to compare translations against the gold standard. Other studies on MT use other metrics such as inter-annotator agreement, but this would be difficult to implement in our case. We also record the average time taken in increments of 10 minutes.

By design, the framework components involving the crowd were required to be done consecutively. Some experiments involved examining tradeoffs between quality and payment amount; running these experiments at the same time could introduce bias. To counter this, we staggered the start times of the crowd-based tasks when different values or amounts were being considered.

We devised a process to automatically list ranking tasks on AMT once all segments were translated and when edits were done. We started the clock when a task was listed on AMT and end when all tasks for that step were completed. We sum the time taken for all tasks. The automated steps of disassembly (step 2) and reassembly (step 5) are assumed to be done instantaneously.

To determine the highest-ranked segment by the crowd in the ranking steps (steps 4 and 7), we use a Borda count (Saari, 1999). Borda counts take a rank of p candidate segments; each candidate ranked highest will receive p points, those ranked second will receive $p-1$ points, etc. These counts are summed across all r ranked lists to obtain a total count, with the highest total score selected as the best translation. When p and r are small, Borda counts may lead to ties, so a plurality vote (the largest number of first-place votes) and anti-plurality votes (i.e., the smallest number of last-place votes), in that order, serve as tiebreakers. Indeed, this method resolved all ties in rankings made by the crowd.

TABLE III
THE EFFECTS OF VARYING THE NUMBER OF CROWDWORKERS FOR EACH TASK ON BLEU SCORES AND TIME TAKEN (IN HOURS)
FOR THE PRE-EDITING AND POST-EDITING STEP IN OUR FRAMEWORK TASK, BROKEN OUT BY LANGUAGE.

			Amt Paid Per Trans Doc	French to English				Chinese to English			
				Pre-editing (step 6)		Post-editing (step 8)		Pre-editing (step 6)		Post-editing (step 8)	
				BLEU	Time	BLEU	Time	BLEU	Time	BLEU	Time
Varying the Number of Translators, Editors and Rankers											
Trans	Ed	Rank									
3	1	1	\$26.00	21.77	7:40	36.64	17:10	21.30	8:10	32.92	17:50
3	3	1	\$34.00	21.77	7:40	37.79	17:40	21.30	8:10	36.09	19:20
3	3	3	\$42.00	24.05	8:50	38.04	19:30	22.89	9:30	36.39	20:20
3	5	3	\$50.00	24.05	8:50	39.45	20:30	22.89	9:30	37.11	20:40
5	5	3	\$54.00	23.98	10:10	39.45	22:00	24.59	10:20	37.11	21:30
5	5	5	\$58.00	23.98	10:30	39.45	22:20	24.59	10:40	37.11	21:50

TABLE II
THE EFFECTS OF VARYING THE SIZE AND NUMBER OF SEGMENTS ON BLEU SCORES AND TIME TAKEN (IN HOURS)
FOR THE PRE-EDITING AND POST-EDITING STEP IN OUR FRAMEWORK TASK, BROKEN OUT BY LANGUAGE.

	Amt Paid Per Trans Doc	French to English				Chinese to English				
		Pre-editing (step 6)		Post-editing (step 8)		Pre-editing (step 6)		Post-editing (step 8)		
		BLEU	Time	BLEU	Time	BLEU	Time	BLEU	Time	
Varying the Size and Number of Segments Translated										
2 segments, avg. 1149 words ea.	\$24.00	23.62	5:20	37.28	14:40	22.49	5:40	36.11	18:00	
4 segments, avg. 575 words ea.	\$30.00	23.77	6:50	38.01	16:30	22.73	6:40	36.43	18:10	
8 segments, avg. 287 words ea.	\$42.00	23.76	8:00	38.12	18:10	22.81	8:40	36.56	19:30	
12 segments, avg. 192 words ea.	\$54.00	24.05	8:50	38.04	19:30	22.89	9:30	36.39	20:20	

C. Gold Standard and Baselines

The translation for each transcript provided in the parallel corpora is used as our gold standard. Also, we asked two professional translators to translate each transcript to English at the average market translation rate of \$0.30/word. We asked the translators to provide the time it took to translate each transcript, and we average the times they provided in our analysis. To translate the three transcripts from the two languages, the total cost for professional translation is \$4137.60, representing an average hourly rate of \$36.94. We use this as our first baseline.

In many translation studies involving non-experts, a single worker is asked to translate an entire document or transcript. We wish to see how this larger task would compare to our framework's use of microtasks. Crowdworkers were asked to translate documents from French and Chinese to English. We hired three crowdworkers for each language pair and asked each to translate a single transcript for \$40.00 per transcript (\$0.017/word). We used the Upwork website, which provides task requesters a method to track the actual time workers spent working on their tasks. We paid a total translation cost of \$240.00 (excluding the Upwork platform fees), which calculates to an average wage of \$5.46/hr. We use this as our second baseline.

D. Participants

Participants were hired from AMT. They were required to have an overall approval rate of 90%. Translators were shown the entire transcript in either French or Chinese; the text segment they were asked to translate was marked. Editors were shown the entire document with each segment identified and asked to make the combined document flow smoothly, paying careful attention to tone and flow. Rankers were shown between 3 and 12 segments (depending on the experimental conditions) and asked to rank the segments from best to worst. If a crowdworker did not complete the task according to the rules, we removed the results and relisted the task. We had only a single case where the task had to be repeated.

5 EXPERIMENTS

A. Experiment 1: The Effects of Payment Amount

On average, each transcript is divided into 12 segments of 192 words each. We wished to see if the amount we offered affected the quality and time. Using the three transcripts, we varied the payment amounts presented to crowdworkers offering \$0.25, \$0.50, and \$1.00 to translate one segment (step 3). We offered another set of crowdworkers the same amounts to edit documents for context and flow (step 6). For the steps involving ranking, we paid a consistent \$0.25 each and had three crowdworkers rank each item (in steps 4 and 7). We

TABLE IV
A COMPARISON OF OUR FRAMEWORK'S RESULTS WITH OUR TWO BASELINES. WE COMPARE THE AVERAGE AND STANDARD DEVIATION (SD) OF BLEU SCORE, AND THE AVERAGE TIME TAKEN, BY LANGUAGE.

	Amt Paid Per Trans Doc	French to English			Chinese to English		
		Avg BLEU	SD BLEU	Avg Time	Avg BLEU	SD BLEU	Avg Time
Professional Translator Baseline	\$689.60	38.23	7.09	13:10	36.70	5.63	13:30
Single Crowdsourcer Baseline	\$40.00	33.35	8.53	17:00	31.84	7.73	17:40
Framework	\$50.00	39.45	3.56	20:30	37.11	3.28	20:40

evaluate the documents at two different points, just after the document is reassembled with each highest-ranked segment in step 6 (but before editing), and at when the final translation has been completed at step 8. Table 1 provides an evaluation of each payment amount on the average time and quality.

From Part 1 of Table 1, there is only a marginal difference in quality (as determined by BLEU score) regardless of the amount we pay the crowdworkers for translation and editing. This backs up the findings of Mason and Watts (2010) on crowdworker payments for other types of tasks. We also notice that there is a jump in the BLEU score between the pre-editing step and the post-editing step irrespective of the amount paid, demonstrating the merits of the editing step in our framework. From a task requester's perspective, offering \$0.50 for translation and editing seems to be an appropriate tradeoff between payment and quality; we carry this forward into our following experiments.

The amount of payment offered does affect the task completion time. This makes sense; some crowdworkers have minimum amounts they will accept before engaging in a task. Completion times for Chinese-to-English translations took more time than those from French to English; this is likely an artifact of the relatively smaller number of Chinese translators available on AMT.

In Part 2 of this experiment, we hold payments for translation of each segment and editing of each document constant at \$0.50 and evaluate the effects of the payment amount for ranking on quality and time. For each transcript, we offer crowdworkers the payment amounts of \$0.10, \$0.25, and \$0.50 to rank (steps 5 and 7). The results of ranking payment amounts on the average time, and quality for each language are provided in Part 2 of Table 1.

The BLEU score does not change much in the ranking steps even when we offer five times as much compensation. Although the time taken decreases when we pay more for ratings, the reduction in time is not as sensitive to cost as it is for editing and translations. We keep payments for rankings at \$0.25 for follow-on experiments as this seems to provide the best balance between quality and payment.

B. Experiment 2: The Effects of Segment Size

Larger segment sizes provide a single translator with more context to work with, which is likely to improve translation quality. On the other hand, smaller segment sizes are more

suitable to crowdsourcing platforms, which are designed for *microtasks*, or small tasks that can be accomplished quickly and inexpensively. To examine the effects of segment size, we divide up each of the three transcripts into segment sizes of approximately 1000 words, 500 words, and 250 words each, representing 2, 4 and eight segments per transcript. We increase the payments for translation (step 3) to correspond with the increase in task size (recall we paid \$0.50 for each of 12 translated segments of 192 words in Experiment 1); corresponding payments are \$3.00, \$1.50, and \$0.75 per segment. We pay a constant \$0.50 for editing the translated text and \$0.25 for ranking. The results are shown in Table 2.

From Table 2, using larger segment sizes decreases the time taken, but overall it has little effect on the quality of the resulting translation. This seemed puzzling at first glance, but it shows the value of the framework's editing and ranking steps. We also note that having more segments reduced the variance in quality as well, although this is not shown in the table. We notice that the best quality is achieved with eight segments and we carry this forward in our next experiment.

C. Experiment 3: The Effects of the Number of Crowdworkers

We have used three crowdworkers at each step in the previous experiments – would increasing or decreasing the number of crowdworkers used in each step influence quality and time taken? We vary the number of crowdworkers as shown in Table 3.

Quality is most sensitive to the number of editors and least sensitive to the number of rankers. We notice there is no benefit to increasing the number of rankers from three to five. There is also no benefit to increasing the number of translators beyond three. The translation step is the most sensitive to the amount of time taken, is the bottleneck in our framework, and is the only step dependent on bi- and multi-lingual crowdworkers; adding additional translators slows the overall translation process down.

6 ANALYSIS

Using a framework design with three translators, five editors, and three rankers, which provides a balance between payment amount and quality, we compare our results with those obtained by the other baselines. Table 4 shows this comparison.

Our framework achieved a BLEU score that exceeds the professional translations for both languages to English at a fourteenth of the cost. However, our framework approach took nearly 1.5 times as long to complete as the quicker of the two baseline approaches. Our framework also achieved a BLEU score that was convincingly better than the single crowdworker baseline, but as with the professional translator, this comes with the tradeoff of requiring much more time. Additionally, the translation cost using our framework was triple that of the single crowdworker; with quality as our framework's paramount consideration, paying more for better quality is an appropriate tradeoff. Our second experiment had shown that, even when our framework used larger segment sizes (approximating the single crowdworker), it was superior in quality to the single crowdworker based on the BLEU scores. This demonstrates the value of a divide-and-conquer approach used by the framework and how it helps low-quality inputs from malicious crowdworkers and spammers from affecting quality.

One often-stated benefit of crowdwork is that tasks can be done quickly with sufficient quality and at a low price. For the metrics of cost, quality and time, our experiments show it is a challenge to achieve all three simultaneously. Our framework is best designed for those with a focus on cost and quality at the expense of time. The single crowdworker model would cost less to implement and take far less time, but our experiments show that quality is significantly inferior. The professional translator would work best when quality and time are important, but the cost is not an issue.

We have been able to examine the time and quality at different points of our framework, demonstrating the verifiability principle. The standard deviation for the crowdworker baseline approach is more than double that of our framework, demonstrating the framework's robustness. It also means our results are not dependent on only a small subset of crowdworkers delivering quality outputs and obscuring the spam from the remaining crowdworkers, which is a common problem with crowdwork.

7 CONCLUSION AND FUTURE WORK

We have introduced a framework for implementing expert quality for translations that approach professional quality at a lower cost of hiring a professional. This framework extends the work of many previous researchers but adds checks for robustness, verifiability, and consistency. Proponents of crowdwork often state the benefits are quality work that is cheaper and quicker than work provided by experts. Our experiments have shown that only two of the three are achievable; using a single crowdworker provides lower quality, a professional translator is higher cost, and our framework takes significantly more time. Expert quality can be achieved using our framework and as few as three translators and five document editors for editing, along with three other crowdworkers to rank the outputs from each of these steps.

Using this configuration, we obtained higher quality than a professional at about a sixth of the cost. The tradeoff is that the configuration required more time than a professional translator or a single translator hired from a crowdsourcing platform.

In future work, we plan to examine how low resource languages might affect quality, cost and time taken by crowdworkers. Our work focused on using AMT as our crowdsourcing platform, but there are many other platforms now available which support entirely different demographics. Evaluating a variety of platforms will allow us to examine the robustness principle in greater detail. We also plan to examine how to better integrate MT tools in our framework to determine if we can improve the quality at an even lower cost. We also plan to examine hybrid approaches like those proposed by Borromeo et al. (2016). These authors performed some initial translations using Google Translate and then asked crowdworkers to make edits to those translations. This could be easily incorporated into our framework. We expect this approach to lower costs and decrease completion time while maintaining high quality in our framework as well. We also plan on examining the work of Gao et al. (2015). They have devised a method to stop once they receive a translation that meets a minimum score threshold. We could incorporate this into our proposed framework, and we expect this would also lower costs.

It may be surprising to note that the professional translators did not achieve better BLEU scores than the framework we introduced. This may be an artifact of the source of the TED 2013 corpora, which used volunteer transcriptions and translations from the TED web site. To ensure the translation quality was at the professional level, we had translators unaffiliated with this study examine the translations, and they verified they were of professional quality. This may call into suspicion the BLEU metric as a fair assessment tool, which is beyond the scope of this study.

Our framework could also apply to other NLP tasks such as text summarization. We are currently conducting experiments to examine if our framework can ensure robustness, verifiability, and consistency in text summarizations just as it does for translations. Initial findings are positive.

REFERENCES

- [1] Ahmet Aker, Mahmoud El-Haj, M-Dyaa Albakour, and Udo Kruschwitz. Assessing Crowdsourcing Quality through Objective Tasks. In *8th Language Resources and Evaluation (LREC 2012)*, Istanbul, Turkey 2012.
- [2] George A. Akerlof. "The market for "lemons": Quality uncertainty and the market mechanism." In *Uncertainty in Economics*, pp. 235-251. 1978.
- [3] Michael Bloodgood and Chris Callison-Burch. "Using Mechanical Turk to build machine translation evaluation sets." In *Proceedings of the NAACL HLT 2010 Workshop on Creating Speech and Language Data with Amazon's Mechanical Turk*, pp. 208-211. ACL 2010.

- [4] Ria Mae Borromeo, Maha Alsayasneh, Sihem Amer-Yahia, and Vincent Leroy. Hybrid Deployment Strategies for Crowdsourced Text Creation Tasks. In *Technical Report, University of Grenoble-Alps*, 2016.
- [5] Olivia Buzek, Philip Resnik, and Benjamin B. Bederson. "Error driven paraphrase annotation using mechanical turk." In *Proceedings of the NAACL HLT 2010 Workshop on Creating Speech and Language Data with Amazon's Mechanical Turk*, pp. 217-221. ACL, 2010.
- [6] Chris Callison-Burch. "Fast, cheap, and creative: evaluating translation quality using Amazon's Mechanical Turk." In *Proceedings of the 2009 Conference on Empirical Methods in Natural Language Processing: Volume 1*, pp. 286-295. ACL, 2009.
- [7] Mahmoud El-Haj, Paul Rayson, Scott Piao, and Stephen Wattam. "Creating and validating multilingual semantic representations for six languages: expert versus non-expert crowds." In *Proceedings of the 15th Conference of the European Chapter of the Association for Computational Linguistics*. ACL, p. 61-71. Valencia, Spain 2017
- [8] Ulrich Germann, Michael Jahr, Kevin Knight, Daniel Marcu, and Kenji Yamada. "Fast Decoding and optimal decoding for machine translation." In *Proceedings of the 39th Annual Meeting on Association for Computational Linguistics*, pp. 228-235. ACL, 2001
- [9] Dan Gillick and Yang Liu. "Non-expert evaluation of summarization systems is risky." In *Proceedings of the NAACL HLT 2010 Workshop on Creating Speech and Language Data with Amazon's Mechanical Turk*, pp. 148-151. ACL, 2010.
- [10] Mingkun Gao, Wei Xu, and Chris Callison-Burch. "Cost optimization in crowdsourcing translation: Low-cost translations made even cheaper." In *Proceedings of the 2015 Conference of the North American Chapter of the Association for Computational Linguistics: Human Language Technologies*, pp. 705-713. 2015.
- [11] Shinsuke Goto, Donghui Lin, and Toru Ishida. "Crowdsourcing for Evaluating Machine Translation Quality." In *10th Language Resources and Evaluation (LREC 2014)*, pp. 3456-3463. 2014.
- [12] Christopher G. Harris and Tao Xu. 2011. The importance of visual context clues in multimedia translation. In *International Conference of the Cross-Language Evaluation Forum for European Languages (CLEF 2011)* (pp. 107-118). Springer Berlin Heidelberg.
- [13] Leah Hoffmann. 2009. Crowd control. *Communications of the ACM* 52(3) pp 16-17. DOI: <http://dx.doi.org/10.1145/1467247.1467254>
- [14] Juan Pablo Hourcade and Laura Gehrt. "Crowdsourcing for delivering research results to patients." In *Proceedings of HCI Korea*, pp. 196-202. Hanbit Media, Inc., 2014.
- [15] Aniket Kittur, Ed H. Chi, and Bongwon Suh. "Crowdsourcing user studies with Mechanical Turk." In *Proceedings of the SIGCHI conference on human factors in computing systems*, pp. 453-456. ACM, 2008.
- [16] Winter Mason, and Duncan J. Watts. "Financial incentives and the performance of crowds." *ACM SigKDD Explorations Newsletter* 11, no. 2 (2010): 100-108.
- [17] Scott Novotney and Chris Callison-Burch. "Shared task: crowdsourced accessibility elicitation of Wikipedia articles." In *Proceedings of the NAACL HLT 2010 Workshop on Creating Speech and Language Data with Amazon's Mechanical Turk*. ACL, 2010.
- [18] Kishore Papineni, Salim Roukos, Todd Ward, and Wei-Jing Zhu. "BLEU: a method for automatic evaluation of machine translation." In *Proceedings of Computational linguistics*, pp. 311-318. Association for Computational Linguistics, 2002.
- [19] Donald G. Saari. "Explaining all three-alternative voting outcomes." *Journal of Economic Theory*, 87, no. 2 (1999): 313-355.
- [20] Rion Snow, Brendan O'Connor, Daniel Jurafsky, and Andrew Y. Ng. "Cheap and fast---but is it good? Evaluating non-expert annotations for natural language tasks." In *Proceedings of empirical methods in natural language processing*, pp. 254-263. ACL, 2008.
- [21] May Al Sohibani, Najd Al Osaimi, Reem Al Ehaidib, Sarah Al Muhanna, and Ajantha Dahanayake. "Factors that influence the quality of crowdsourcing." In *New Trends in Database and Information Systems II*, pp. 287-300. Springer, Cham, 2015.
- [22] Jörg Tiedemann, 2012, Parallel Data, Tools and Interfaces in OPUS. In *Proceedings of the 8th International Conference on Language Resources and Evaluation (LREC 2012)*
- [23] Yonghui Wu, Mike Schuster, Zhifeng Chen, Quoc V. Le, Mohammad Norouzi, Wolfgang Macherey, Maxim Krikun, et al. "Google's neural machine translation system: Bridging the gap between human and machine translation." *arXiv preprint arXiv:1609.08144* (2016).
- [24] Rui Yan, Mingkun Gao, Ellie Pavlick, and Chris Callison-Burch. "Are Two Heads Better than One? Crowdsourced Translation via a Two-Step Collaboration of Non-Professional Translators and Editors." In *ACL (1)*, pp. 1134-1144. 2014.
- [25] Omar F. Zaidan and Chris Callison-Burch. "Crowdsourcing translation: Professional quality from non-professionals." In *Proceedings of the 49th Annual Meeting of the Association for Computational Linguistics: Human Language Technologies-Volume 1*, pp. 1220-1229. ACL, 2011.

Sports Video Anonymisation and Accuracy Prediction

César Marrades Cortés, Keith Quille, Jelena Vasić, and Seán McHugh

Abstract—The anonymisation of people featuring in videos is required in many contexts. One of these is the physical education state exam in Ireland, where secondary school students are assessed as prescribed by the National Council for Curriculum and Assessment (NCCA), with the use of video recordings among other tools. For reasons of GDPR, all the video material presented for grading must not reveal the identity of the students. The work presented here was undertaken on consultation with the NCCA regarding their needs in this area and comprised two distinct tasks: (1) the implementation and testing of an anonymiser program in C#, supported by libraries Accord and EmguCV, each defining a different variant of the program; (2) the use of machine learning predictive models in Weka to investigate which of various factors (such as camera quality, camera angle, sport) affect the anonymisation program's performance on sports videos. One hundred video inputs, resulting in 200 outputs (one for each of the two libraries, per input), were used and the best anonymisation success prediction model had an accuracy of 94% and a specificity of 95.2%. This work forms a base upon which a full automated video anonymisation system could be built, most importantly generating knowledge on what measures can be taken towards the optimisation of video anonymisation performance.

Index Terms—Video anonymisation, machine learning.

I. INTRODUCTION

THE WORK presented here addresses a specific image processing problem presented by the Irish National Council for Curriculum and Assessment (NCCA). Physical education exams are often recorded by students in video format and then evaluated by the NCCA. The project aimed to develop a face recognition system that demonstrates the possibility of automatically anonymising the videos prior to evaluation, by means of using face recognition libraries to detect students' faces and of blurring them. This is an important concern for the NCCA as identity preservation is required by the current European General Data Protection Rules (GDPR). It is also a way of avoiding bias during content evaluation.

The paper presents findings and recommendations in two distinct, if related, areas: (1) the development and integration of a software system prototype, with emphasis on facial recognition libraries used and (2) a data-analytical

investigation of the capabilities and limitations of those libraries, including what features of video material make it most amenable to processing for anonymisation.

The libraries used in the work at hand have been employed by other researchers. In her master's thesis [1], Suad Hajr Ahmed Omar discusses the general limitations of the Emugcv library in terms of an existing optimal camera-to-subject distance of 1 to 3 feet, required illumination and problems with recognition when the face is partially covered, e.g. with sunglasses. Team TBD [2] found that in a situation with time constraints and related performance requirements the Emgucv library presented as the most stable and complete solution. Mohammed [3] also used the Emugcv library, while recognising that several factors can limit its usability, for example unusual poses and facial expressions or occlusion by sunglasses or make-up.

II. METHODOLOGY

The prototype development and performance analysis phases of the work are logically and operationally separate, each self-contained with respect to methodology.

A. Prototype Development

The main functional requirement placed on the prototype was that it accept a video clip of a sporting activity as input and that it output a video based on the original clip, in which the faces of any people are rendered unrecognisable.

The prototype was created as a Microsoft Windows executable file, developed using C# with the Emgucv and Accord face recognition libraries. These libraries were chosen based on favourable reports in literature on similar uses and availability of documentation (Accord), in combination with their availability at no monetary cost. As the focus of the project was to generically investigate automated anonymisation for a particular type of video rather than provide an exhaustive review of implementation possibilities, it was sufficient to use these two libraries. Other libraries considered were Microsoft Cognitive Services, Censor Face and OpenCV.

The prototype was designed to generate two outputs, one produced by each of the face recognition libraries, for every received input video. It transforms all the video files found in a specified input folder.

Manuscript received on June 12, 2019, accepted for publication on September 7, 2019, published on December 30, 2019.

The authors are with Technological University Dublin, Ireland (e-mail: cmarrades@hotmail.com, {keith.quille, jelena.vasic, sean.mchugh}@tudublin.ie).

TABLE I
SPORTS: DETAILS OF SPORTS THAT FEATURE IN THE SET OF INPUT VIDEOS,
TOGETHER WITH CO-OCCURRING CONDITIONS (OTHER VARIABLE VALUES).

Sport	Conditions
Basketball	indoors and outdoors, dynamic cameras, medium and far distances
Karate, Judo	indoors, multiple backgrounds, multiple angles, tendency to close to medium distances, head protectors
Boxing	indoors, multiple angles, head protectors, referee, lose, medium and far ranges
Hurling	tendency towards far distances, high angles, with multiple people
Gaelic Football	tendency towards far distances, medium and high angles, with multiple people
Tennis	multiple angles, mixed cam movement types, high angles
Table Tennis	medium and far distances
Soccer, Soccer Interior	multiple angles, mixed cam movement types, medium, high angles
High Jump, High Jump Pole	medium angles from close and medium distances
Weight Lifting	different camera types static cameras from close and medium distances and medium angles

B. Analysis

The data used in the analysis phase comprised input and output videos and information about the videos, which was extracted in a manual process.

Videos of various sports activities, in mp4 format, were downloaded from YouTube to serve as inputs to the anonymiser prototype. They were chosen so as to provide a variety of different scenarios covering the most common cases handled by the NCCA when evaluating students in exams. Longer videos require longer processing time, therefore shorter videos had a preference over longer ones.

The attributes of both the input and the output videos were extracted 'manually', by observation and noting of properties on the part of a researcher. This manual processing of the 300 videos took about 7 days. The attributes extracted were the following:

- Sport (sports defined in Table I)
- Indoors/outdoors
- Light conditions
- Whether the video was recorded using a phone or a dedicated camera
- Video quality
- Camera movement e.g. static or dynamic
- Camera recording angle e.g. low, medium or high
- Camera distance e.g. medium or far

- Number of people in the video (ranging from 1 to more than 10)
- Whether people were wearing head protectors
- Field type e.g. court, grass
- Field colour
- Background colour
- Whether there was crowd on the background of the video
- Whether there was a referee
- Ground position: whether people in the video spend a considerable amount of time on the floor e.g. with wrestling or combat sports in general

Table I lists the sports that feature in the input video set. It also shows co-occurring conditions i.e. other attribute values that tend to pertain to videos of the listed sport. The ten sports have been chosen to cover a wide variety of scenarios i.e. attribute combinations and are not based on a definite list from the NCCA.

The output videos were labelled according to two different scales for measuring the quality of the blurring: a 1-10 scale that rates outputs from worst to best and a binary one, with values of *full* and *partial* for the level of facial blurring. The different values (classes) on these scales are shown in Table II.

The following rules were used when grading outputs using the classifications in Table II:

- *no accuracy*: no faces were recognised or blurred

TABLE II
OUTPUT CLASSIFICATION: VALUES AND THEIR DESCRIPTIONS
FOR THE TWO TARGET VARIABLES.

10-Value Scale Class	10-Class Description	2-Value Scale Class
1	no accuracy, high noise	Partial
2	no accuracy, noise	Partial
3	extremely low accuracy, noise (any)	Partial
4	low accuracy, high noise	Partial
5	low accuracy, noise	Partial
6	good accuracy, high noise	Partial
7	good accuracy, low noise	Full
8	almost no failures, noise (any)	Full
9	no failures, noise	Full
10	no failures, no noise	Full

- *extremely low accuracy*: almost no faces were recognised or blurred
- *low accuracy*: some of the faces were recognised in some parts of the video
- *good accuracy*: the majority of faces were recognised and blurred
- *almost no failures*: very few faces not recognised or blurred
- *no failures*: all faces were recognised and fully blurred

The Weka tool was used for all the steps required to develop the prediction model i.e. for data pre-processing, attribute selection and the selection of a final model that would be used to predict the quality of the outputs produced by the anonymiser.

III. RESULTS

Explicit measurements were not taken of the anonymiser prototype's performance but the following account provides insight into the relevant orders of magnitude. The other two subsections respectively contain a detailed discussion of the videos and their attributes and of the predictive models that were built in Weka.

A. Running the Prototype

The anonymiser prototype was used to create 200 output videos from 100 videos collected to serve as input. A micro Amazon Web Services (AWS) instance was initially set up to process the 100 inputs. This instance was later scaled to a large instance as the process was using the CPU at maximum capacity. After a few days, a second Intel Core i7 2.60GHz machine was added to speed up the process. Processing all the 100 inputs and producing 200 outputs took around 8 days following these steps.

B. Video Attributes

The input videos were viewed and attribute values, as listed above, recorded for each of them. The output videos were viewed and the blurring quality assessed using the scales presented in Table II. Thus the data set for analysis was completed.

The initial prediction model defined in Weka used the 10-value target variable shown in Table II and performed poorly, with a maximum accuracy of 56%. For this reason a second, binary target variable with values *Full* and *Partial*, denoting 'fully blurred' and 'partially blurred', was defined as shown in Table II. The best accuracy achieved when predicting this new target was 94%. Although no further tests were done, final results indicate that increasing the *Full* threshold to rank 8 would still have high probability of positive results.

Data inspection and observations revealed that videos are more or less evenly divided between indoors and outdoors. Only 5% of the videos are recorded under bad light conditions and there is a high tendency to record these videos with cameras rather than with mobile phones, as only 5% of them are recorded using a phone. There are very few videos where persons are wearing head protections and there is a tendency to keep the same cam distance for the duration of the video. Recordings are evenly divided between having and not having a referee and similarly for the presence of a crowd.

Very few output videos scored highly for successful facial recognition and blurring. No obvious correlations were identified by visual correlation inspection. The distributions show that *fully* blurred outputs have very specific characteristics, while the *partially* blurred ones constitute most of the data set, with only 10% of the set scoring above 6. The plots indicate that all the *fully* blurred outputs are taken from close distances and tend to be recorded from static or 'steady' cameras, from a medium angle, with no crowd. These attributes will be important in determining whether a video will be successfully anonymised.

Figure 1 shows the class distributions for the 10-class and 2-class target labelling.

TABLE III
MOST RELEVANT ATTRIBUTES.

Attribute	Rank
Crowd	0.3242
CamDistance	0.3203
In-Out	0.3203
CamMovement	0.2958
CamAngle	0.2695
Referee	0.261

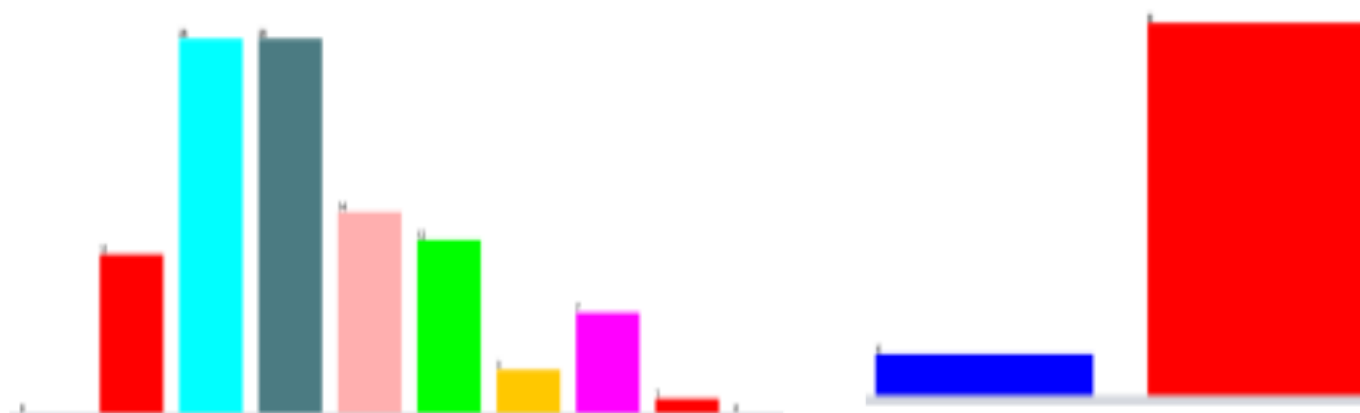


Fig. 1. Output video class distribution with 10-class labelling (left, classes 1 to 10 shown from left to right) and 2-class labelling (right, blue = fully blurred, red = partially blurred).

TABLE IV
PREDICTIVE MODEL PERFORMANCE.

Attribute Set	Algorithm	Accuracy	Sensitivity	Specificity
1	Naïve Bayes	91%	100%	90%
1	Trees.J48	89%	30%	95.5%
2	Naïve Bayes	94%	90%	94.4%
2	Trees.J48	89%	30%	95.5%

Weka correlation attribute evaluation found *Crowd*, *Camera distance*, *Indoors/outdoors* and *Camera movement* as the most relevant attributes affecting video blurring accuracy. A longer list of relevant attributes and their ranks is given in Table III.

C. Predictive Models

The attribute set in Table III, chosen using Weka correlation attribute evaluation on the dataset with the 2-class target, was used in two variants, with *Referee* (attribute set 1) and without *Referee* (attribute set 2), each to build two different predictive model types. The performance of the 4 resulting models is presented in Table IV, in terms of accuracy, sensitivity and specificity. Accuracy represents overall predictive power, while sensitivity and specificity respectively measure the power of predicting the *fully blurred* and *partially blurred* classes.

IV. EVALUATION AND CONCLUSION

The work described in this paper investigates solutions to two different interrelated problems: the automated anonymisation of sports videos and the evaluation of the anonymisation process. The anonymisation prototype was successfully built and run on a batch of a 100 sports videos downloaded from the Web to produce anonymised output videos. The properties of the input and output videos were summarised and the video attributes most predictive of successful anonymisation identified.

A. Anonymiser Prototype

The prototype is a functioning piece of software, however, the many parameters that informed its development, in particular the face recognition libraries, and its eventual performance characteristics both in terms of speed and anonymisation success, demonstrate that automating the removal of person-identifying visual information from videos is not a trivial task.

For example, the processing of 100 videos took 8 days on two high-end processors. This could be improved upon through code parallelisation.

The face recognition libraries used have particular limitations, including difficulty with identifying faces obstructed with arms or head-wear such as hats and sunglasses or poor performance with unfocussed or blurred footage. In this sense the prototype is a very specific implementation of the generic functionality of interest. However, its main value is in the proof of concept, not only in the functional sense but also as it serves as a kind of a 'wet-lab' for the analytic work presented here, which heavily relies on it.

B. Anonymisation Performance Prediction

A large number of video attributes, some specifically pertinent to sports videos, were investigated and tested for relevance to the success of face detection in a video. For example, the presence of a crowd, camera distance and indoor vs. outdoor filming affect the success of anonymisation. This kind of information is helpful when recommendations can

be made in the production of video material for a particular purpose such as examination.

The idea of automated extraction of attribute values, which in this instance had to be identified 'manually' from the videos, while secondary to the work at hand, would have wide application in video-related research. Closer to the central topic of this paper, further work could continue on to solidify and generalise the conclusions through investigation across a greater number of face identification libraries and video types.

REFERENCES

- [1] H. Suad and O. Ahmed, "Biometric system based on face recognition system," Master's dissertation, Firat University, 2016.
- [2] P. Bergeron, K. Kuchta, M. Olenik, B. Stern, and C. Tosswill, "Kodak Facebook collage project," Rochester IoT, Tech. Rep., 2016.
- [3] S. Abdo and A. Mohammed, "Recognition of face detection system based on vide," Universiti Teknikal Malaysia Melaka, Tech. Rep., 2015.

Marching Cubes Algorithm for Transforming Images

Angélica Hernández Rayas, Delia Irazú Hernández Farías, Eduardo Pérez Careta, Rafael Guzmán Cabrera, José Francisco Gómez-Aguilar, José Zacarías Huamaní, and Teodoro Cordova Fraga

Abstract—A Marching Cubes algorithm modification is presented in this work. This proposal is used to make from surface image discretized on volumetric pixels, called voxels, a three-dimensional representation from a flat image is raised. To achieve this effect, each component is connected continuously (without gaps) regardless of the values of the image surface. The aim is to achieve this effect, optimizing computational resources. The ambiguous elimination cases as well as the identification of non-trivial cases are among the main features attended. Once identified, the modelling process is optimized, seeking congruence with the edges of the volume, applying techniques that allow connecting the edges adjacent to the corners, generating the triangulation of the cube, and introducing a new set of edge equivalence classes and cube faces. The obtained results allow to observe the effective performance of the implemented modification.

Index terms—Image processing, marching cubes algorithm, surface reconstruction.

1. INTRODUCTION

THREE-DIMENSIONAL surface reconstruction is an important topic in various application areas, such as quality inspection and reverse engineering. The goal of image-based 3D reconstruction is to infer the 3D geometry and structure of objects and scenes from one or multiple 2D images. Many image-based reconstruction methods have been proposed, based on photometric as well as geometric principles. For example, in [1] and [2] the authors used ray

casting to find the surface and shading it with hue gradients or grayscales. A limitation of these ray casting algorithms is that they result in approximate shading with an unnormalized gradient. In [3], the authors present an approximation started on the contour surface and progressively connected triangles on the surface between adjacent slices. In [4], the authors proposed a network, which takes as input an RGB image and a target viewpoint. Other techniques estimate multiple depths maps from pre-defined or arbitrary viewpoints, for example, in [5] and [6] the authors followed the same approach but predict the depth maps, along with their binary masks, from pre-defined viewpoints. In [7], additional vertices are added to account for these sharp features. However, a subsurface with more than one component would cause ambiguities for connecting contours.

The standard surface reconstruction algorithm is Marching Cubes (henceforth MC). In the original MC algorithm, sharp edges create undesirable artifacts. The simplest technique is to draw the voxels directly on the screen with a simple 2D square. When the MC algorithm was first published, the fact that there were a few ambiguities in some of the vertex configurations was not completely known. The basic principle of the MC algorithm [8] is to subdivide space into a series of small squares, i.e., to sample the space on a regular grid. This algorithm examines the data in groups of eight, with each group forming a small cube. For each corner of the cube, the algorithm decides whether it is on the inside or the outside by using the provided region definitions, see Fig. 1.

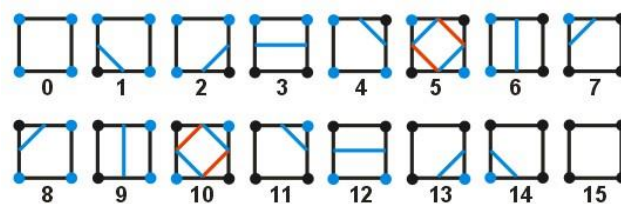


Fig. 1. The 16 square combinations of the MC algorithm.

When we process a single cube, from the sequence of cubes, we first classify each of the eight corner vertices of the cube identified as positive values if they are within the image surface and as negative values if they are outside the

Manuscript received on July 8, 2019, accepted for publication on September 17, 2019, published on December 30, 2019.

Angélica Hernández Rayas, Delia Irazú Hernández Farías, José Zacarías Huamaní, and Teodoro Cordova Fraga are with División de Ciencias e Ingenierías, Universidad de Guanajuato Campus León, Loma del Bosque 103, Lomas del Campestre, CP 37150, León, GTO, Mexico.

Eduardo Pérez Careta and Rafael Guzmán Cabrera are with División de Ingenierías, Universidad de Guanajuato Campus Irapuato-Salamanca, Carretera Salamanca-Valle de Santiago km 3.5 + 1.8, Comunidad de Palo Blanco, 36885, Salamanca, GTO, Mexico.

José Francisco Gómez-Aguilar is with CONACyT-Tecnológico Nacional de México / CENIDET, Interior Internado Palmira S/N, Col. Palmira, 62490, Cuernavaca, MOR, Mexico.

Corresponding author: Rafael Guzmán Cabrera (e-mail: guzmanc@ugto.mx)

TABLE I
THE MC ALGORITHM RECOGNIZED
15 EQUIVALENCE CLASSES.

#0 (2)	#1 (16)	#2 (24)
#3 (24)	#4 (8)	#5 (48)
#6 (48)	#7 (16)	#8 (24)
#9 (6)	#10 (2)	#11 (6)
#12 (12)	#13 (12)	#14 (8)

image surface. The vertices belonging to the internal triangulation of a cube when they are at the edge or one end can be computationally classified as: "outside" or empty space, and on the other hand "inside" of the image. The classifications vary between "inside / outside" successively of the eight corners of the cube. There are exactly 256 different possible combinations for each cube. The original MC algorithm represents these 256 cases, 28 in 15 equivalent cases, as shown in Table 1. Two cases can remain in the same class when we rotate a cube around an axis that passes through the center so that it can coincide with another case exactly.

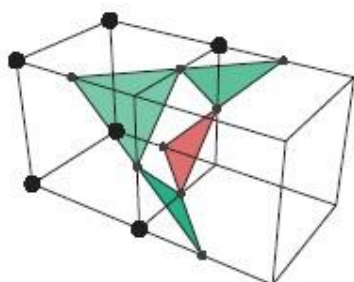


Fig. 2. Two cubes presenting an ambiguous face.

As shown in Table 1, the number in the lower-left corner of each entry is the class index, and the number in the lower right corner is the number of cases that belong to the equivalence class of the 256 total cases. A black dot in the corner indicates that it is within the volume of the solid and the corner without a black dot indicates that it is outside the volume of the solid. Green triangles indicate that they are the front concerning the point of view and red triangles indicate that they are hidden faces behind the cube material. If we choose to enforce a case described in Table 1, then problems are created in some situations with two cubes that share the same face, this can be ignored in some occasions, but if not, one or the other has to be inverted cube for proper image formation using the "inside/outside" option according to the cube situation. There is a result of a mismatch between two cubes that generates a kind of rectangle hole in the final triangulation of the two cubes, in Figure 2 it can be observed that the cube on the left must be inverted to match the respective equivalency of the image cell. The edges of the resulting triangulations do not fit along the shared face and a large rectangular hole is created. Green polygons are located on the outside of the cube and red polygons are located on the inside of the cube.

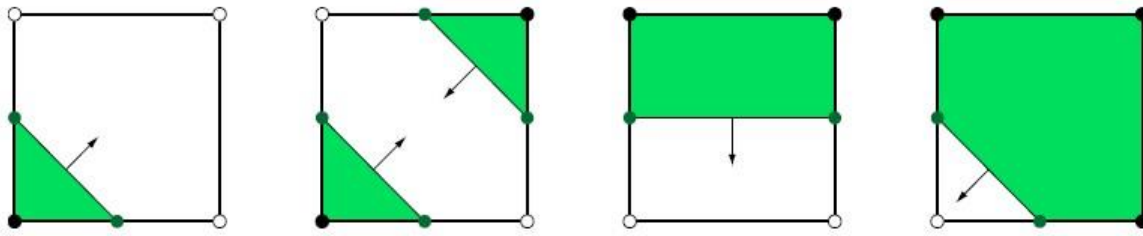


Fig. 3. The image volume edge congruence requirement

2. PROPOSED CHANGES TO THE MARCHING CUBES ALGORITHM.

When generating the image surface based on voxels, our most important interest is that the resulting surface is accurate, that is, each component is connected continuously, without gaps, regardless of the values of the image surface. Assuming that this requirement is achieved, our main objective among the remaining precautions is a maximum performance using the minimum computational resource to obtain results in short times. The main objective of the algorithm modifications' is to eliminate ambiguous cases without adding calculation per cube, to achieve this, the cases are considering the possible computational states in Table 1, it is divided into two groups: "inside / outside" out of the four corners of a face of a given cube, there are only four non-trivial cases after we eliminated the ones that are equivalent through the rotation.

The image volume edge congruence requirement can be satisfied when there is an ambiguous case as long as we connect vertices at adjacent edges that share a bottom corner, see Figure 3, they are the only non-trivial edge settings allowed by our algorithm (and a fifth configuration is a trivial face that has no edge because the four corner faces have the same state inside or outside).

Our modified MC algorithm does not allow the edge configuration, and therefore the triangulation of the left cube in Figure 2 is not considered to be a contaminant element. This circumstance requires the introduction of a new set of triangulation equivalency classes that require edge-by-face configuration, as shown in Figure 3, the only four non-trivial edge configurations that can be introduced in the face of a cube by the modification of the MC. This fact is due to a rotation of the cube. Vertices with solid points are classified inside and vertices with points are classified as outside. The green region represents solid space, and the arrows represent the normal surface direction outward along the edges.

Can be solved if the inversion of the equivalence relation is done for the classes that have an ambiguous face, leaving only the rotation. Next, it is set up three new equivalence classes to contain cases that have more than four interior corners in the image volume from the original equivalence class #2, #6, and #7 shown in Table 2. The result is the set

of the 18 equivalence classes shown, where three new classes are shown in the last row. Classes #15, #16, and #17 are made up of the inverse representation of cases #2, #6, and #7, and their respective rotations.

The number in the lower-left corner of each entry is the class index, and the number in the lower right corner is the number of cases that belong to the equivalence class of the 256 total cases. A black dot in the corner indicates that it is within the volume of the image, and the corner without a dot indicates that it is outside. Green triangles are in front of the cube face and red triangles are behind the cube material ions.

If chooses to enforce a case described in Table 1, then problems are created in some situations with two cubes that share the same face, this can be ignored in some occasions, but if not, one or the other cube has to be inverted to the adequate formation of the image using the "inside/outside" option according to the situation of the cube. There is a result of a mismatch between two cubes that generates a kind of rectangle hole in the final triangulation of the two cubes as shown in Figure 4.

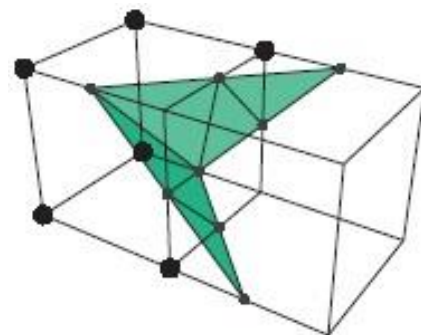


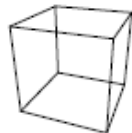
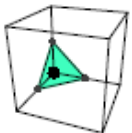
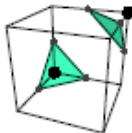
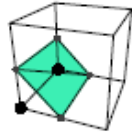
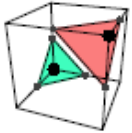
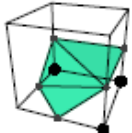
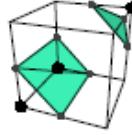
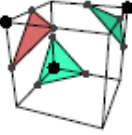
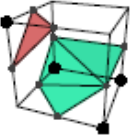
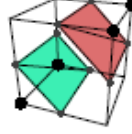
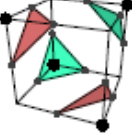
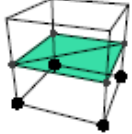
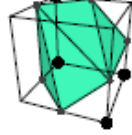
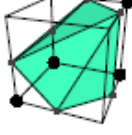
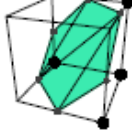
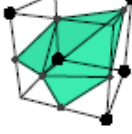
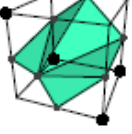
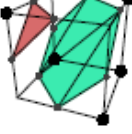
Fig. 4. Two cubes sharing an ambiguous face.
The left side belongs to one of the new equivalence classes

A. Marching Cube Algorithm

Modifications to the Marching Cube algorithm were designed considering the following points:

- Possibility to edit small surface regions without rebuilding the whole image surface.

TABLE II
MC ALGORITHM WITH 18 EQUIVALENCE CLASSES IDENTIFIED.

 #0 (2)	 #1 (16)	 #2 (12)
 #3 (24)	 #4 (8)	 #5 (48)
 #6 (24)	 #7 (8)	 #8 (24)
 #9 (6)	 #10 (2)	 #11 (6)
 #12 (12)	 #13 (12)	 #14 (8)
 #15 (12)	 #16 (24)	 #17 (8)

- The previous presentation of the surface to show the details of the image.
- The resulting mesh is defined with no imperfections between triangles or regions without the rise that arose the surface.
- Vertices it is shared as efficiency between each neighboring cube and take full advantage of the hardware to transform artifacts and decrease the storage requirements of the resulting surface.
- Minimize the use of memory and computational resources by generating meshing on the image surface.
- Dividing the total image into relatively small voxels of equal size helps to achieve these goals.

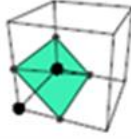
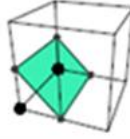
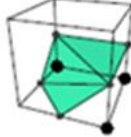
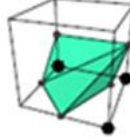
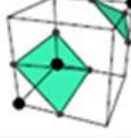
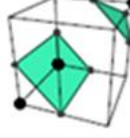

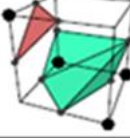
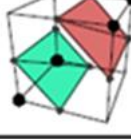
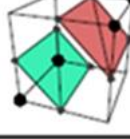
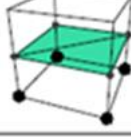
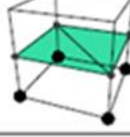
So that the level of detail of analysis of the total volume of the solid is high, blocks of $16 \times 16 \times 16$ voxels are divided for their respective analysis. Each block has a coordinate axis regardless of the division it is, and these are x, y, and z from one to any block joint set.

3. RESULTS

Table 3(a) shows the 12 equivalence classes for which multiple triangulations are possible, the two triangulations shown are the only ones that test us for the better division at high-resolution voxel data, they are chosen to be as different as possible in terms of their main directions of curvature, the second column, represent the total triangulations are possible for each equivalence class.

Since the lowest levels of detail are normally processed only when the pixel intensity is low, the triangulation we assign to any cube only makes a subtle difference that can be observed directly on the surface mesh. However, there are two indirect ways in which the ability to choose between triangulations makes an important difference. The first way is to increase the level of detail to a few sets of cubes on the surface, visualizing a noticeable "jump" that occurs in the mesh when the level of detail is abruptly changed, but the visible magnitude of this transition is

TABLE III
THE 12 EQUIVALENCE CLASSES FOR WHICH
MULTIPLE TRIANGULATION: POSSIBLE -1.

#3			$C_2 = 2$
#5			$C_3 = 5$
#6			$C_2 = 2$
#8			$C_3 = 5$
#9			$C_2^2 = 4$
#11			$C_2 = 2$

reduced when the low resolution of the triangulation is a better division for the high-resolution surface .

Table 4 shows appearance of shading is the possibility -2. A very noticeable artifact can appear when a poor choice of triangulation is made because it can create a concavity that produces unwanted shadow when aligned in a way unfavorable to the surface view. Choosing the best triangulation from Tables 3 and 4 helps eliminate these defects by creating a smoother low-resolution surface mesh.

We refer to the meaning of transition to be one of the forms shown in Figure 5, which is influenced by nine empty voxels on one side. The face opposite the full resolution face is called the half resolution face, and the sample values at its four corners are identical to that of the corresponding corners of the full resolution face. The width of a transition cube (that is, the distance between full resolution and average face resolution) is a freely adjustable parameter on the image surface.

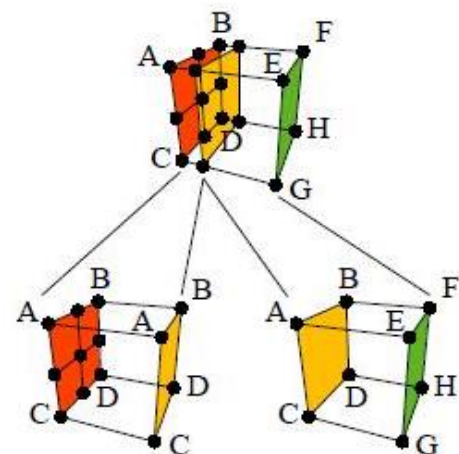

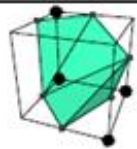
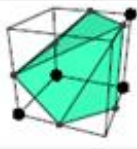
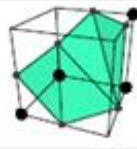
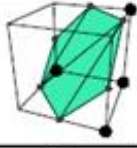
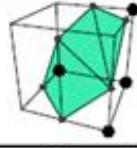


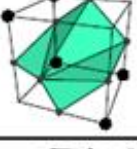
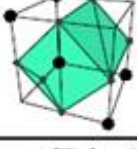
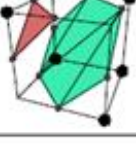
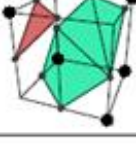


Fig. 5. A transition cube is divided into two parts along a plane parallel to the boundary face between maximum resolution block and medium resolution

TABLE IV
THE 12 EQUIVALENCE CLASSES FOR WHICH
MULTIPLE TRIANGULATIONS: POSSIBLE -2

#12			$C_4 = 14$
#13			$C_4 = 14$
#14			$C_4 = 14$
#15			$C_4 = 14$
#16			$C_5 = 42$
#17			$C_4 = 14$

Our new algorithm triangulates the left side using sample values that only appear on the full resolution side. The right part is triangulated with the conventional modified MC algorithm, see Figure 5.

4. CONCLUSION

The transition cubes always have the same configuration regarding the number and location of our voxels, and this is true even when a medium resolution block is bordered by full resolution of two- or three-part blocks. An equivalence class is a trivial class that contains the two cases in which the interior or exterior state of the nine sample values is the same.

The modifications proposed to the MC algorithm in this work allow us to overcome some of the deficiencies that the original algorithm has, allowing us to make the recovery of surfaces without gaps, which allows us to obtain a 3D representation of a flat image in a more efficient.

ACKNOWLEDGMENTS

Authors wishing to acknowledge at the University of Guanajuato for the partial support in this project under grant DAIP2019/59023. Also José Francisco Gómez Aguilar acknowledges the support provided by CONACyT: Cátedras CONACyT para jóvenes investigadores 2014.

REFERENCES

- [1] E. J. Farrell. Color display and interactive interpretation of three-dimensional data. IBM J. Res. Develop 27, pp 13-19, August 1983.
- [2] K. H. Hohne and R. Bernstein. Shading 3D images from CT using gray-level gradients. IEEE Transactions On Medical Imaging MI-5, pp. 45-47, March 1986.
- [3] E. Keppel. Approximating complex surfaces by triangulation of contour lines. IBM J. Res. Develop 19, pp. 2-11, January 1975.
- [4] M. Tatarchenko, A. Dosovitskiy, and T. Brox, "Multi-view 3D models from single images with a convolutional network," in ECCV, 2016, pp. 322–337.

- [5] A. A. Soltani, H. Huang, J. Wu, T. D. Kulkarni, and J. B. Tenenbaum, "Synthesizing 3D shapes via modeling multi-view depth maps and silhouettes with deep generative networks," in IEEE CVPR, 2017, pp. 1511–1519.
- [6] C.-H. Lin, C. Kong, and S. Lucey, "Learning Efficient Point Cloud Generation for Dense 3D Object Reconstruction," AAAI, 2018.
- [7] L. Kobbelt, M. Botsch, U. Schwanerke, and H.-P. Seidel. Feature sensitive surface extraction from volume data. In Proceedings of SIGGRAPH 2001, pp. 57-66, 2001
- [8] W. E. Lorensen and H. E. Cline, *Marching Cubes: A High Resolution 3D Surface Construction Algorithm*, Computer Graphics 21(4): 163-169, 1987.

Towards Recognizing Phrase Translation Processes: Experiments on English-French

Yuming Zhai, Pooyan Safari, Gabriel Illouz, Alexandre Allauzen, and Anne Vilnat

Abstract—When translating phrases (words or group of words), human translators, consciously or not, resort to different translation processes apart from the literal translation, such as **Idiom Equivalence, Generalization, Particularization, Semantic Modulation, etc.** Translators and linguists (such as Vinay and Darbelnet, Newmark, *etc.*) have proposed several typologies to characterize the different translation processes. However, to the best of our knowledge, there has not been effort to automatically classify these fine-grained translation processes. Recently, an English-French parallel corpus of TED Talks has been manually annotated with translation process categories, along with established annotation guidelines. Based on these annotated examples, we propose an automatic classification of translation processes at subsentential level. Experimental results show that we can distinguish non-literal translation from literal translation with an accuracy of 87.09%, and 55.20% for classifying among five non-literal translation processes. This work demonstrates that it is possible to automatically classify translation processes. Even with a small amount of annotated examples, our experiments show the directions that we can follow in future work. One of our long term objectives is leveraging this automatic classification to better control paraphrase extraction from bilingual parallel corpora.

Index Terms—Translation processes, non-literal translation, automatic classification.

I. INTRODUCTION

SINCE 1958, translators and linguists have published work on translation processes [1], [2], [3], [4]. They distinguish literal translations from other translation processes at subsentential level. Consider these two human non-literal translation examples: the first translation preserves exactly the meaning, where the fixed expression *à la hauteur de* ‘to the height of’ has a figurative sense which means *capable of solving*; while the second one is more complicated, there exists a textual inference between the source text and the translation.

(1.EN) *a solution that’s big enough to solve our problems*
 (1.FR) *une solution à la hauteur de nos problèmes*
 (2.EN) *and that scar has stayed with him for his entire life*
 (2.FR) *et que, toute sa vie, il a souffert de ce traumatisme*
 (‘he has suffered from this traumatism’)

Non-literal translations can bring difficulties for automatic word alignment [5], [6], or cause meaning changes in certain

cases. However, to the best of our knowledge, there has not been effort to automatically classify these fine-grained translation processes to benefit downstream natural language processing tasks. For example, Machine Translation (MT) techniques have been leveraged for paraphrase extraction from bilingual parallel corpora [7], [8]. The assumption is that two monolingual segments are potential paraphrases if they share common translations in another language. Currently the largest paraphrase resource, PPDB (ParaPhrase DataBase) [9], has been built following this method. Nonetheless, Pavlick *et al.* [10] revealed that there exist other relations (*i.e. Entailment (in two directions), Exclusion, Other related and Independent*)¹ than strict equivalence (paraphrase) in PPDB. Non-literal pivot translations inside the parallel corpora could break the strict equivalence between the candidate paraphrases extracted, whereas they have not received enough attention during this corpora exploitation.

From a linguistic point of view, apart from the word-for-word literal translation, different versions of human translations reflect the richness of human language expressions, where various translation processes could be employed. Furthermore, because of the existing differences between languages and cultures, non-literal translation processes are sometimes inevitable to produce correct and natural translations. The fine-grained phrase-level translation processes could help foreign language learners to better compare the language being learned with another language already mastered.

Based on the theories developed in translation studies and through manually annotating and analyzing an English-French parallel corpus, Zhai *et al.* [11] have proposed a typology of translation processes adapted to their corpus. In this work our main contribution is proposing an automatic classification of translation processes at subsentential level, based on these annotated examples. From the aspect of granularity and our goal of better controlling paraphrasing process or helping foreign language learners, it is different from the task of filtering semantically divergent parallel sentence pairs to improve the performance of MT systems [12], [13], [14]. Experimental results show that we can distinguish non-literal translation processes from literal translation with an accuracy of 87.09%, and 55.20% for classifying among non-literal multi-classes.

¹Exclusion: X is the contrary of Y; X is mutually exclusive with Y. Other related: X is related in some other way to Y. (*e.g. country / patriotic*). Independent: X is not related to Y.

Manuscript received on June 10, 2019, accepted for publication on September 7, 2019, published on December 30, 2019.

The authors are with LIMSI-CNRS, Univ. Paris-Sud, Univ. Paris-Saclay, France (e-mail: {firstname.lastname}@limsi.fr).

The first two authors contributed equally to this article.

In the present paper, after reviewing related work, we describe the manual annotation and the data set. Exploited features and different neural network architectures will be presented, followed by experimental results and error analysis. Finally we conclude and present the perspectives of this work.

II. RELATED WORK

Translators and linguists have proposed several typologies to characterize different translation processes. Vinay and Darbelnet [1] identified direct and oblique translation processes, the latter being employed when a literal translation is unacceptable, or when structural or conceptual asymmetries arising between the source language and the target language are non-negligible. Following studies include, among others, the work of Newmark [2], [15], Chuquet and Paillard [3]. More recently, Molina and Hurtado Albir [4] proposed their own categorization based on studying the translation of cultural elements in the novel *A Hundred Years of Solitude* from Spanish to Arabic.

Non-literal translations or cross-language divergences have been studied to improve MT related techniques. In order to enable more accurate word-level alignment, Dorr *et al.* [5] proposed to transform English sentence structure to more closely resemble another language. A translation literalness measure was proposed to select appropriate sentences or phrases for automatically constructing MT knowledge [16]. Using a hierarchically aligned parallel treebank, Deng and Xue [6] semi-automatically identify, categorize and quantify seven types of translation divergences between Chinese and English.² Based on the syntactic and semantic similarity between bilingual sentences, Carl and Schaeffer [17] developed a metric of translation literality. We have drawn inspiration from these preceding work for our feature engineering.

Recently, different models have been proposed to automatically detect translation divergence in parallel corpora, with the goal of automatically filtering out divergent sentence pairs to improve MT systems' performance. An SVM-based cross-lingual divergence detector was introduced [12], using word alignments and sentence length features. Their following work [13] proposed a Deep Neural Network-based approach. This system could be trained for any parallel corpus without any manual annotation. They confirmed that these divergences are a source of performance degradation in neural machine translation. Pham *et al.* [14] built cross-lingual sentence embeddings according to the word similarity with a neural architecture in an unsupervised way. They measure the semantic equivalence of a sentence pair to decide whether to filter it out.

Another task studying human translations concerns automatic post-editing [18]. The aim is evaluating systems

for automatically correcting translation errors of an unknown "black box" MT engine, by learning from human revisions of translations produced by the same engine. Evaluation metrics include TER [19], BLEU [20] and manual evaluation. The task that we propose here is different from these attempts, which either filter semantically divergent sentence pairs to improve the performance of MT systems; or automatically correct machine translation errors to improve the translation quality. Our task of classifying translation processes (in two classes or in multi-classes) at subsentential level is a stand-alone task. One of our long term objectives is leveraging this automatic classification to better control phrase-level paraphrase extraction from bilingual parallel corpora.

III. MANUAL ANNOTATION AND DATA DESCRIPTION

In order to model translation choices made by human translators at subsentential level, Zhai *et al.* [11] have annotated a trilingual parallel (English-French, English-Chinese) corpus of TED Talks³ with translation processes. The corpus is composed of transcriptions and human translations of oral presentations. The inter-annotator agreement (Cohen's Kappa) [21] for annotating the English-French and English-Chinese control corpus is 0.67 and 0.61, both around the substantial agreement threshold. This indicates that the task of manual annotation is already complicated. Readers can find more details of corpus construction in the article [11].

The automatic classification is conducted on the English-French pair in this work. We present in the table I a brief definition, a typical example and the number of instances for each category to be automatically classified.⁴ We combine *Transposition* and *Mod+Trans* in a category *Contain_Transposition*, where *Modulation* is considered as a neutral part. We will work on the classification of the pair English-Chinese once the annotation phase is finished. In this work, we conduct experiments in a simplified scenario, where we already know the boundaries of bilingual pairs, and we only predict the translation process. For example, given the pair *deceptive* → *une illusion* in a pair of bilingual sentences, the goal is to predict its label *Contain_Transposition*.

IV. AUTOMATIC CLASSIFICATION

We have tried two approaches for the automatic classification. Since the size of the cross validation data set is quite small, we first compare different statistical machine learning techniques with feature engineering. We also build different neural network architectures which we explain below.

A. Feature Engineering with Statistical Machine Learning Techniques

We describe below the features exploited in this work. The tag sets of English and French for part-of-speech (PoS)

²Lexical encoding; difference in transitivity; absence of language-specific function words; difference in phrase types; difference in word order; dropped elements; structural paraphrases.

³<https://www.ted.com/>

⁴Note that there are other detailed annotation rules in the annotation guidelines.

TABLE I

DEFINITION, TYPICAL EXAMPLE AND NUMBER OF INSTANCES FOR EACH TRANSLATION PROCESS TO BE AUTOMATICALLY CLASSIFIED. THE INSTANCES WERE MANUALLY ANNOTATED IN AN ENGLISH-FRENCH PARALLEL CORPUS OF TED TALKS. WE COMBINE *Transposition* AND *Mod+Trans* IN A CATEGORY *Contain_Transposition* FOR THE AUTOMATIC CLASSIFICATION.

Translation Process	Definition and typical example
Literal (3771)	Word-for-word translation, also concerns lexical units in multiword form. <i>certain kinds of</i> → <i>certain types de</i>
Equivalence (289)	Non-literal translation of proverbs or fixed expressions; a word-for-word translation makes sense but the translator expresses differently, without changing the meaning and the grammatical classes. <i>back then</i> → <i>à l'époque</i> ('at that time')
Generalization (86)	Several source words or expressions could be translated into a more general target word or expression, the translator uses the latter to translate. <i>as we sit here in ...</i> → <i>alors que nous sommes à ...</i> ('as we are at ...')
Particularization (215)	The source word or expression could be translated into several target words or expressions with a more specific meaning, and the translator chooses one of them according to the context. <i>the idea I want to put out is ...</i> → <i>l'idée que je veux diffuser c'est ...</i> ('the idea I want to spread is ...')
Modulation (195)	Metonymical and grammatical modulation [3]; change the point of view; the meaning could be changed. <i>that scar has stayed with him</i> → <i>il a souffert de ce traumatisme</i> ('he has suffered from this traumatism')
Transposition (289)	Change grammatical classes without changing the meaning. <i>unless something changes</i> → <i>à moins qu'un changement ait lieu</i> ('unless a change occurs')
Mod+Trans (53)	Combine the transformations of <i>Modulation</i> and of <i>Transposition</i> , which could make the alignment difficult. <i>this is a completely unsustainable pattern</i> → <i>il est absolument impossible de continuer sur cette tendance</i> ('it is completely impossible to continue on this trend')

tagging, constituency parsing and dependency parsing have been converted into three compact and unified tag sets [22].

1) The PoS tagging is done by *Stanford CoreNLP* [23] for the two languages. On source and target side, for each PoS tag, the number of its occurrence is counted in a vector. We also calculate the cosine similarity between these two vectors (on all words and only on content words).⁵

2) We verify the pattern of PoS tag sequence changing according to a manual list, for example the pair *methodologically* → *de façon méthodologique* 'methodologically' corresponds to the pattern *ADV* → *ADP NOUN ADJ*.

3) The number of tokens in the two segments (l_e , l_f), the ratio of these numbers (l_e/l_f , l_f/l_e), the distance Levenshtein [24] between the segments.

4) The constituency parsing is done by *Bonsai* [25] for French, by *Stanford CoreNLP* for English. We compare the PoS tags for a pair of words, the non-terminal node tags for a pair of segments, the tag category (e.g. verb → verb phrase) for a word translated by a segment or vice versa.

5) The dependency parsing is done by *Stanford CoreNLP* for the two languages. Inside the segments, the number of occurrence of each dependency relation is counted. Outside the segments, among the words linked at source and target side, we filter those which are aligned in the sentence context. Then the number of occurrence of each dependency relation

⁵The tags of content words include: ADJ, ADV, NOUN, PROP, VERB. If a segment does not contain any content word, the original segment is used.

between the words in segments and these context words is counted.

6) The cosine similarity is calculated between the embeddings from *ConceptNet Numberbatch* [26]. This resource is multilingual and the system based on *ConceptNet* took the first place in the task "Multilingual and Cross-lingual Semantic Word Similarity" of SemEval2017 [27], [28]. Certain multi-word expressions have their own embeddings in this resource. Otherwise, we calculate the average of embeddings only on content words. The same features are calculated for lemmatized segments.⁶

7) The resource *ConceptNet* [26] also provides assertions in triplet: a pair of words or expressions linked by a relation. In this multilingual resource, we verify if an English-French pair is directly linked; indirectly linked by another French segment or simply not linked.⁷ Three forms are tested: original form, lemmatized form and lemmatized filtered form.⁸

8) On the lemmatized filtered form, we calculate the percentage of tokens which are linked with a relation of derivation, based on the resource *ConceptNet*. For example *deceptive* and *illusion* 'illusion' are not directly linked in the resource, but they are both linked to *illusoire* 'illusory'. Hence

⁶The lemmatization is done by *Stanford CoreNLP* and *Tree Tagger* [29] for English and French.

⁷The EN-FR and FR-FR assertions are used in this work.

⁸We filter the words in a manual list, for example the light verbs, determinants, pronouns, etc.

we consider that there exists a link of derivation between them.

For the three following features, we have exploited the lexical translation probability table generated by the statistical word alignment tool *Berkeley Word Aligner* [30], trained on an English-French parallel corpus composed of TED Talks and a part of Paracrawl corpus (in total 1.8M parallel sentence pairs and 41M English tokens).⁹

9) The entropy of the distributions of lexical translation probabilities [31], [17], calculated according to this equation: $H(X) = \sum_i P(x_i) I(x_i) = -\sum_i P(x_i) \log_e P(x_i)$. We calculate the average entropy on content words. A bigger entropy indicates that the words have more general meanings or they are polysemous. The same feature is calculated on the lemmatized content words.

10) The bidirectional lexical weighting on content words, by supposing a n - m alignment a between the segments $(\bar{e}$ and \bar{f}). In the scheme proposed by Koehn *et al.* [32] (equation 1), to calculate the direct lexical weighting, each of the English words e_i is generated by aligned foreign words f_j with the word translation probability $w(e_i|f_j)$. And similarly for the reverse lexical weighting $lex(\bar{f}|\bar{e}, a)$. The same feature is calculated for lemmatized content words. This feature could reflect the alignment confidence between a pair of segments.

$$lex(\bar{e}|\bar{f}, a) = \prod_{i=1}^{length(\bar{e})} \frac{1}{|\{j|(i, j) \in a\}|} \sum_{\forall (i, j) \in a} w(e_i|f_j) \quad (1)$$

11) The sum of lexical translation probability differences between the human translation and the most probable translation according to the probability table. For each source word, we take the target word in human translation with the biggest probability. According to this method, we also count the unaligned words to calculate a ratio on the total number of tokens on each side. These features are calculated in the two directions of translation.

We use the toolkit *Scikit-Learn* [33] to train different statistical machine learning classifiers.¹⁰

B. End-to-end Neural Network Architectures

The source and target phrases are encoded using a bidirectional encoder with Gated Recurrent Unit (GRU) (size 10). The outputs of forward and backward recurrent networks are concatenated to form the source and target phrase representations (size 20). After the encoder layer we have tried two different architectures. The first one is to build an alignment matrix for the source-target phrases, using the dot product of the two representations, inspired by these two work [34], [14]. Then a *Convolutional Neural Network* (CNN) classifier is applied to this alignment matrix, which is composed of one convolution layer followed by pooling.

⁹<https://wit3.fbk.eu/>, <https://paracrawl.eu/index.html>

¹⁰The code and data set is publicly available at <https://github.com/YumingZHAI/ctp>.

Since the shape of the alignment matrix varies from one source-target pair to another, an adaptive pooling is used [35]. The output of the pooling layer is fed into a fully-connected layer followed by a linear layer as the output. In the second architecture, the source and target outputs of the encoder layer are averaged over time steps to produce two fixed-dimensional vectors, which are further concatenated (size 40) and fed into a Multi-layer Perceptron (MLP) classifier. The hidden layer of MLP includes 10 hidden units with tanh non-linearity.

The length of our phrases is usually short, especially for word-for-word *Literal* instances. In order to build a more robust alignment matrix and to avoid the out-of-vocabulary problem, we finally choose to use character embeddings. As shown in table II, for the embedding layer, we have tried respectively randomly initialized character embeddings (size 10), and training our own word embeddings using skipgram model of *FastText* [36] on a TED Talks corpus (around 3M tokens for both English and French), with a word-embedding size of 100, minimum n-gram of 3, and maximum n-gram of 6. All the models have been trained in 200 epochs, with a learning rate of 0.0001 using Adam optimizer and the minibatch size of 20. Dropout has been applied to all layers except the output and embedding layers.

V. EXPERIMENTAL RESULTS AND ANALYSIS

Table II and III show the results of our classifiers using end-to-end neural network architectures, for binary classification (balanced distribution) and multi-class classification. For the binary classification, *Non_literal* (NL) class has in total 1127 instances, and 1127 *Literal* (L) instances are randomly chosen. Besides the preprocessing steps of lowercasing and correcting minor spelling errors, for the neural classifiers, we also normalized the clitic forms to complete words (*e.g.* 're → are), and normalized digits to letter form (*e.g.* 42 → four two). The architecture using word embeddings and MLP obtain better results and is faster than the other two architectures. However, the current data set is too small for neural architectures to produce satisfactory results.

TABLE II
BINARY CLASSIFICATION
(BALANCED DISTRIBUTION)

Architecture	Accuracy	F1 (L)	F1 (NL)
Randomly initialized character embedding			
CNN	59.99%	0.60	0.60
MLP	71.16%	0.71	0.71
Pre-trained fasttext word embedding			
MLP	71.25%	0.71	0.71

The number of all non-literal instances (1127) is only one third of *Literal* instances (3771). Considering this important difference, for the statistical machine learning classifiers, we first evaluated them under these configurations:

- six classes (*Literal*, *Equivalence*, *Generalization*, *Particularization*, *Modulation*, *Contain_Transposition*). We first put

TABLE III
MULTI-CLASS CLASSIFICATION
(FIVE NON-LITERAL CLASSES)

Architecture	Accuracy	Micro-F1	Macro-F1
Randomly initialized character embedding			
CNN	34.08%	0.34	0.20
MLP	40.74%	0.41	0.34
Pre-trained fasttext word embedding			
MLP	43.22%	0.43	0.34

all *Literal* instances. Then to have an approximately balanced class distribution, we randomly take 200 instances for *Literal*.

- two classes (*Literal* and *Non_literal*), with three distributions (3:1, 2:1, 1:1). The distribution 3:1 is the natural distribution in the data set. The instances of *Literal* have been extracted randomly for the last two distributions.

- five classes (only non-literal categories).

For each configuration, we have tuned the hyperparameters of different classifiers. We evaluate them by five-fold cross-validation,¹¹ using the metrics such as the average accuracy of five folds, the micro average and macro average F1-score [37]. The *DummyClassifier* is used as a baseline, which generates random predictions by respecting the distribution of training classes.

First, we attempted a direct classification into six classes (see table IV). The best results by *RandomForest* reflect the difficulty of the task in multi-classes. On the other hand, we observe the potential of our features on classifying the category *Literal* when the number of instances increases. As a result, we decide to divide the problem: conduct first a binary classification, and secondly a multi-class classification among the non-literal categories.

For the binary classification, the two best classifiers are *RandomForest* and MLP. Furthermore, *RandomForest* has better performance than the two combined by the method *hard voting* or *soft voting*. The table IV presents the results under three different class distributions. From the natural distribution (3:1) to our artificial balanced distribution by randomly choosing *Literal* instances (thus both class have 1127 instances), the average F1-score for the class *Non_literal* increases from 0.78 to 0.88. We will continue to test this tendency when a larger data set is available. Table IV also shows the results for the classification into five non-literal classes using all features, and the average F1-score for each non-literal category are shown in table V. The category *Generalization* has many fewer instances than the other categories, which need to be augmented; there exist many confusions between *Modulation* and the other categories, which suggests rather a review of annotation guidelines.

Table VI recapitulates the best performance on binary classification (balanced distribution) and on the classification of five non-literal classes, using the most helpful set of features. With the best performing classifier *RandomForest*, we

have investigated the performance of features one by one and also grouped them: *PoS_tagging* (feature 1, 2), *surface* (feature 3), *syntactic_analysis* (feature 4, 5), *external_resource* (feature 6, 7, 8) and *word_alignment* (feature 9, 10, 11). For binary classification, feature 10 (bidirectional lexical weighting) is most helpful, which generates average F1-score of 0.78 for *Literal* and 0.80 for *Non_literal* by itself. The group of features *word_alignment* contributes the most for the binary classification. The combination of all features generates the best results, which remain the same if we remove the feature 4 (constituency parsing), 7 (how the pair is linked in the resource *ConceptNet*) and the features on PoS tagging apart from the vector counting the occurrence of each tag. The features in float form generally perform better than those in discrete form (e.g. 0, 1, etc.). Concerning the classification into five non-literal classes, the combination of all features except the group *external_resource* leads to the best results, where the group *PoS_tagging* and *syntactic_analysis* contribute more than the group *word_alignment* and *surface*. The accuracy changes from 55.10% to 55.20% after feature ablation (see table IV).

Our error analysis shows that in binary classification, it is difficult to distinguish *Literal* and *Equivalence*; in multi-class classification, the biggest confusion is between *Equivalence* and *Contain_Transposition*. Consequently, we conducted another three binary classification experiments (see table VII), where in all configurations each class has 549 instances to make the results comparable: i) *Literal* vs *Non_literal* ii) *Literal* combined with *Equivalence* (E), vs the other classes iii) *Literal* combined with *Equivalence* and *Transposition* (T), vs the other classes. The third configuration is more interesting, because the group of translation processes *LET* do not bring meaning changes, while the processes *non-LET* could. The results show that by including *Transposition* (change grammatical classes without changing the meaning), the performance gets better than only grouping *Literal* and *Equivalence*, since we avoid the confusion between *Equivalence* and *Transposition*. The better results of binary classification (L vs NL, LET vs non-LET) indicate that in future work we can develop cascading classifiers, namely first separating word-for-word literal translations, or those which do not cause meaning changes, then conducting a finer-grained classification among the other categories.

VI. CONCLUSION AND PERSPECTIVES

We have proposed a new Natural Language Processing task of automatically classifying translation processes at subsentential level, based on manually annotated examples from a parallel English-French TED Talks corpus. To the best of our knowledge, these translation processes have not been explicitly exploited during paraphrase extraction from bilingual parallel corpora. With the best performing classifier *RandomForest* and feature engineering, our empirical results show a best accuracy of 87.09% for

¹¹StratifiedKfold is used for cross-validation, where the folds are made by preserving the percentage of samples for each class.

TABLE IV
CLASSIFICATION RESULTS UNDER DIFFERENT CONFIGURATIONS, USING ALL FEATURES

Distribution of classes	Classifier	Accuracy	Micro-F1	Macro-F1
Six classes				
six classes, with 3771 <i>Literal</i>	Dummy	60.76%	0.61	0.15
	RandomForest	83.10%	0.83	0.44
six classes, with 200 <i>Literal</i>	Dummy	18.92%	0.19	0.16
	RandomForest	57.04%	0.57	0.52
Two classes				
<i>Literal</i> (3) : <i>Non_literal</i> (1)	Dummy	65.84%	0.66	0.52
	RandomForest	90.16%	0.90	0.86
<i>Literal</i> (2) : <i>Non_literal</i> (1)	Dummy	56.43%	0.56	0.51
	RandomForest	88.85%	0.89	0.88
<i>Literal</i> (1) : <i>Non_literal</i> (1)	Dummy	53.19%	0.53	0.53
	RandomForest	87.09%	0.87	0.87
Five classes				
Five non-literal classes	Dummy	20.32%	0.20	0.18
	RandomForest	55.10%	0.55	0.47

TABLE V
AVERAGE F1-SCORE FOR EACH NON-LITERAL CLASS, USING ALL FEATURES

Category	Equivalence	Generalization	Particularization	Modulation	Contain_Transposition
Nb. instances	289	86	215	195	342
Average F1	0.51	0.25	0.56	0.36	0.68

TABLE VI
CLASSIFICATION RESULTS AFTER FEATURE ABLATION STUDY

	average accuracy	average F1-scores	
binary classification (balanced distribution)	87.09%	0.87 (<i>Literal</i>)	0.88 (<i>Non_literal</i>)
five non-literal classes	55.20%	0.55 (micro average)	0.48 (macro average)

TABLE VII
CLASSIFICATION RESULTS AFTER GROUPING CLASSES, EVERY CLASS HAS 549 INSTANCES

Configuration	average accuracy	average F1 (class1)	average F1 (class2)
Dummy	48.63%	0.49	0.49
L vs NL	85.24%	0.84	0.86
LE vs non-LE	75.32%	0.74	0.77
LET vs non-LET	79.42%	0.78	0.81

binary classification (*Literal* vs *Non_literal*) and 55.20% for multi-class classification (*Equivalence*, *Generalization*, *Particularization*, *Modulation*, *Contain_Transposition*), which are much better than the baseline random classifier.

This task is complicated, and our exploratory work is restrained by the limited amount of annotated examples. However, our work demonstrates that automatically classifying translation processes seem possible, and the experiments show the directions that we can follow in future work. There is much room to constitute an augmented and balanced data set, on which we will evaluate our classifier to observe the performance. The finer error analysis of the classification results is useful to help the research on corpus annotation and linguistic analysis. We will continue to improve the classifier on English-French, by implementing other features for multi-class classification, and explore more neural architectures. We will also extend our work to English-Chinese translation pairs. One of our long term objectives is leveraging this automatic classification to better control paraphrase

extraction from bilingual parallel corpora.

REFERENCES

- [1] J.-P. Vinay and J. Darbelnet, *Stylistique comparée du français et de l'anglais: méthode de traduction*, ser. Bibliothèque de stylistique comparée. Didier, 1958.
- [2] P. Newmark, *Approaches to Translation (Language Teaching Methodology Series)*. Oxford: Pergamon Press, 1981.
- [3] H. Chuquet and M. Paillard, *Approche linguistique des problèmes de traduction anglais-français*. Ophrys, 1989.
- [4] L. Molina and A. Hurtado Albir, "Translation techniques revisited: A dynamic and functionalist approach," *Meta*, vol. 47, no. 4, pp. 498–512, 2002.
- [5] B. J. Dorr, L. Pearl, R. Hwa, and N. Habash, "Duster: A method for unraveling cross-language divergences for statistical word-level alignment," in *Conference of the Association for Machine Translation in the Americas*. Springer, 2002, pp. 31–43.
- [6] D. Deng and N. Xue, "Translation divergences in Chinese–English machine translation: An empirical investigation," *Computational Linguistics*, vol. 43, no. 3, pp. 521–565, 2017.
- [7] C. Bannard and C. Callison-Burch, "Paraphrasing with bilingual parallel corpora," in *Proceedings of the 43rd Annual Meeting on Association for Computational Linguistics*. Association for Computational Linguistics, 2005, pp. 597–604.

- [8] J. Mallinson, R. Sennrich, and M. Lapata, "Paraphrasing revisited with neural machine translation," in *Proceedings of the 15th Conference of the European Chapter of the Association for Computational Linguistics: Volume 1, Long Papers*, vol. 1, 2017, pp. 881–893.
- [9] J. Ganitkevitch, B. Van Durme, and C. Callison-Burch, "PPDB: The paraphrase database," in *Proceedings of the 2013 Conference of the North American Chapter of the Association for Computational Linguistics: Human Language Technologies*, 2013, pp. 758–764.
- [10] E. Pavlick, J. Bos, M. Nissim, C. Beller, B. Van Durme, and C. Callison-Burch, "Adding semantics to data-driven paraphrasing," in *Proceedings of the 53rd Annual Meeting of the Association for Computational Linguistics and the 7th International Joint Conference on Natural Language Processing (Volume 1: Long Papers)*, vol. 1, 2015, pp. 1512–1522.
- [11] Y. Zhai, A. Max, and A. Vilnat, "Construction of a multilingual corpus annotated with translation relations," in *First Workshop on Linguistic Resources for Natural Language Processing*, 2018, pp. 102–111.
- [12] M. Carpuat, Y. Vyas, and X. Niu, "Detecting cross-lingual semantic divergence for neural machine translation," in *Proceedings of the First Workshop on Neural Machine Translation*. Association for Computational Linguistics, 2017, pp. 69–79.
- [13] Y. Vyas, X. Niu, and M. Carpuat, "Identifying semantic divergences in parallel text without annotations," in *Proceedings of the 2018 Conference of the North American Chapter of the Association for Computational Linguistics: Human Language Technologies, NAACL-HLT 2018, New Orleans, Louisiana, USA, June 1-6, 2018, Volume 1 (Long Papers)*, M. A. Walker, H. Ji, and A. Stent, Eds. Association for Computational Linguistics, 2018, pp. 1503–1515.
- [14] M. Q. Pham, J. Crego, J. Senellart, and F. Yvon, "Fixing translation divergences in parallel corpora for neural mt," in *Proceedings of the 2018 Conference on Empirical Methods in Natural Language Processing*, 2018, pp. 2967–2973.
- [15] P. Newmark, *A textbook of translation*. Prentice Hall New York, 1988, vol. 66.
- [16] K. Imamura, E. Sumita, and Y. Matsumoto, "Automatic construction of machine translation knowledge using translation literalness," in *Proceedings of the tenth conference on European chapter of the Association for Computational Linguistics-Volume 1*. Association for Computational Linguistics, 2003, pp. 155–162.
- [17] M. Carl and M. J. Schaeffer, "Why translation is difficult: A corpus-based study of non-literality in post-editing and from-scratch translation," *HERMES-Journal of Language and Communication in Business*, no. 56, pp. 43–57, 2017.
- [18] R. Chatterjee, M. Negri, R. Rubino, and M. Turchi, "Findings of the WMT 2018 shared task on automatic post-editing," in *Proceedings of the Third Conference on Machine Translation*. Belgium, Brussels: Association for Computational Linguistics, October 2018.
- [19] M. Snover, B. Dorr, R. Schwartz, L. Micciulla, and J. Makhoul, "A study of translation edit rate with targeted human annotation," in *Proceedings of association for machine translation in the Americas*, vol. 200, no. 6, 2006.
- [20] K. Papineni, S. Roukos, T. Ward, and W.-J. Zhu, "BLEU: a method for automatic evaluation of machine translation," in *Proceedings of the 40th annual meeting on association for computational linguistics*. Association for Computational Linguistics, 2002, pp. 311–318.
- [21] J. Cohen, "A coefficient of agreement for nominal scales," *Educational and Psychological Measurement*, vol. 20, pp. 37–46, 1960.
- [22] S. Petrov, D. Das, and R. T. McDonald, "A universal part-of-speech tagset," in *Proceedings of the Eighth International Conference on Language Resources and Evaluation, LREC 2012, Istanbul, Turkey, May*
- 23–25, 2012, N. Calzolari, K. Choukri, T. Declerck, M. U. Dogan, B. Maegaard, J. Mariani, J. Odijk, and S. Piperidis, Eds. European Language Resources Association (ELRA), 2012, pp. 2089–2096.
- [23] C. D. Manning, M. Surdeanu, J. Bauer, J. Finkel, S. J. Bethard, and D. McClosky, "The Stanford CoreNLP natural language processing toolkit," in *Association for Computational Linguistics (ACL) System Demonstrations*, 2014, pp. 55–60.
- [24] V. I. Levenshtein, "Binary codes capable of correcting deletions, insertions, and reversals," *Soviet physics doklady*, vol. 10, no. 8, pp. 707–710, 1966.
- Association for Computational Linguistics. Chinese Information Processing Society of China, 2010, pp. 108–116.
- [26] R. Speer, J. Chin, and C. Havasi, "Conceptnet 5.5: An open multilingual graph of general knowledge," in *Thirty-First AAAI Conference on Artificial Intelligence*, 2017, pp. 4444–4451.
- [27] J. Camacho-Collados, M. T. Pilehvar, N. Collier, and R. Navigli, "Semeval-2017 task 2: Multilingual and cross-lingual semantic word similarity," in *Proceedings of the 11th International Workshop on Semantic Evaluation (SemEval-2017)*. Association for Computational Linguistics, 2017, pp. 15–26.
- [28] R. Speer and J. Lowry-Duda, "ConceptNet at SemEval-2017 task 2: Extending word embeddings with multilingual relational knowledge," in *Proceedings of the 11th International Workshop on Semantic Evaluation, SemEval@ACL 2017, Vancouver, Canada, August 3-4, 2017*, S. Bethard, M. Carpuat, M. Apidianaki, S. M. Mohammad, D. M. Cer, and D. Jurgens, Eds. Association for Computational Linguistics, 2017, pp. 85–89.
- [29] H. Schmid, "Improvements in part-of-speech tagging with an application to German," in *Proceedings of the ACL SIGDAT-Workshop*, 1995, pp. 47–50.
- [30] P. Liang, B. Taskar, and D. Klein, "Alignment by agreement," in *Proceedings of the main conference on Human Language Technology Conference of the North American Chapter of the Association of Computational Linguistics*. Association for Computational Linguistics, 2006, pp. 104–111.
- [31] R. M. Gray, *Entropy and Information Theory*. Berlin, Heidelberg: Springer-Verlag, 1990.
- [32] P. Koehn, F. J. Och, and D. Marcu, "Statistical phrase-based translation," in *Proceedings of the 2003 Conference of the North American Chapter of the Association for Computational Linguistics on Human Language Technology-Volume 1*. Association for Computational Linguistics, 2003, pp. 48–54.
- [33] F. Pedregosa, G. Varoquaux, A. Gramfort, V. Michel, B. Thirion, O. Grisel, M. Blondel, P. Prettenhofer, R. Weiss, V. Dubourg, J. Vanderplas, A. Passos, D. Cournapeau, M. Brucher, M. Perrot, and E. Duchesnay, "Scikit-learn: Machine learning in Python," *Journal of Machine Learning Research*, vol. 12, pp. 2825–2830, 2011.
- [34] J. Legrand, M. Auli, and R. Collobert, "Neural network-based word alignment through score aggregation," in *Proceedings of the First Conference on Machine Translation, WMT 2016, colocated with ACL 2016, August 11-12, Berlin, Germany*. The Association for Computer Linguistics, 2016, pp. 66–73.
- [35] K. He, X. Zhang, S. Ren, and J. Sun, "Spatial pyramid pooling in deep convolutional networks for visual recognition," *IEEE Transactions on Pattern Analysis and Machine Intelligence*, vol. 37, no. 9, pp. 1904–1916, Sept 2015.
- [36] P. Bojanowski, E. Grave, A. Joulin, and T. Mikolov, "Enriching word vectors with subword information," *Transactions of the Association for Computational Linguistics*, vol. 5, pp. 135–146, 2017.
- [37] G. Tsoumakas, I. Katakis, and I. Vlahavas, "Random k-labelsets for multilabel classification," *IEEE Transactions on Knowledge and Data Engineering*, vol. 23, no. 7, pp. 1079–1089, 2011.

Named Entity Recognition Based on a Graph Structure

David Muñoz, Fernando Pérez, and David Pinto

Abstract—The identification of indirect relationships between texts from different sources makes the task of text mining useful when the goal is to obtain the most valuable information from a set of texts. That is why in the field of information retrieval the correct recognition of named entities plays an important role when extracting valuable information in large amounts of text. Therefore, it is important to propose techniques that improve the NER classifiers in order to achieve the correct recognition of named entities. In this work, a graph structure for storage and enrichment of named entities is proposed. It makes use of synonyms and domain-specific ontologies in the area of computing. The performance of the proposed structure is measured and compared with other NER classifiers in the experiments carried out.

Index Terms—NER, n-grams, text representation, graph-based representation, named entity recognition.

I. INTRODUCTION

THE EXCESSIVE use of computers to produce and manage information around the world has caused an uncontrolled increase in textual information that abounds mainly on the Internet, causing an explosive growth of information overload and resulting very difficult to extract the most valuable information for a specific topic. In order to deal with this problem, there are some proposals such as information retrieval systems, which seek information in a collection of documents and retrieve the most relevant resources based on a specific search [1]. This requires techniques in the process of understanding natural language and is where the recognition of named entities and their effective identification play an important role in information retrieval tasks.

The present research work concerns the construction of a classifier in the task of Named Entity Recognition (NER) based on a data structure represented by a graph with enriched grammatical functions. These functions aim to improve the correct recognition of named entities in order to obtain a better representation of documents and in this way facilitating

some tasks associated with the understanding of texts such as classification of texts and information retrieval.

The technique of storage and enrichment of named entities proposed here is based on a graph structure, where we have nodes and weights in the links that connect to the nodes, and alternative nodes have been added to the original ones based on their synonyms. The alternative routes are created from the synonyms of the parts that make up the entities, thus the classifier can recognize named entities where their components have some relationship with the original entities of the initial corpus. Our proposed NER classifier is compared with other NER classifiers, showing the best results in the Recall measurement but with very poor levels in terms of Precision. It demonstrates an inverse behavior compared to other classifiers, which stand out with a high Precision but a low Recall. Its performance in general through the F1 measure denotes just being below the results obtained by the CRF++ classifier [2]. The results obtained from the classifier are analyzed and discussed.

In summary, this research paper presents a proposal of representation of named entities through a graph structure, exploiting the use of semantic relationships such as synonyms.

The final result obtained is an enriched graph that represents the named entities of a set of documents used as Gold Standard. Thus expanding the correct recognition of named entities in texts.

The rest of this paper is organized as follows. Section 2 shows the state of the art. In section 3, the preliminaries and background related are presented. In section 4, the proposed structure is explained including its construction. Section 5 presents the case study of the classifier, defining the corpus employed, pre-processing of data, measures used to compare the performance as well as the results obtained in comparison with other NER classifiers and a discussion of them. Finally, section 6 ends by showing the contributions made with this research work as well as the mention of future steps.

II. THE STATE OF THE ART

Text Mining [3] is the process of Information Retrieval (IR), Named Entity Recognition and Information Extraction (IE). *Text mining is “The discovery by computer of new, previously unknown information, by automatically extracting information from different written resources”* [4].

Information retrieval is the task of extracting information from a collection of resources of an unstructured nature that satisfies an information need, generally textual data [5].

Manuscript received on May 27, 2019, accepted for publication on September 11, 2019, published on December 30, 2019.

David Muñoz is with the Technological University Dublin, Department of Computing, Ireland, and the Benemérita Universidad Autónoma de Puebla, Facultad de Ciencias de la Computación, Mexico (e-mail: devasc26@gmail.com).

Fernando Pérez is with the Technological University Dublin, Department of Computing, Ireland (e-mail: fernandopt@gmail.com).

David Pinto is with the Benemérita Universidad Autónoma de Puebla, Facultad de Ciencias de la Computación, Mexico (e-mail: davidduardopinto@gmail.com).

In the last years, the task of extracting meaningful data of text has gained the attention in researcher and industry fields [6].

Since information extraction is considered as a limited form of full natural language understanding, where the information we are looking for is known beforehand. It includes two fundamental tasks, Named Entity Recognition and Relation Extraction (RE) [7].

Named Entity Recognition seeks to identify and classify references to named entity mentions in unstructured text into predefined classes [8].

In the task of named entity classification, Mohamed and Oussalah [9] presented an approach by using the Wikipedia article info-boxes where it has significantly reduced the classifier's processing time since the information inside the info-box is structured.

The proposed approach achieved a classification accuracy of above 97% with 3600 named entities and CoNLL-2003 shared task NER dataset used to validate the classifier's performance.

In the task of Named Entity Recognition in Tweets, Ritter et al. [10] proposed a distantly supervised approach which applies Labeled LDA to leverage large amounts of unlabeled data in addition to large dictionaries of entities gathered from Freebase, and combining information about an entity's context across its mentions.

This because classifying named entities in tweets is a difficult task since tweets contain a plethora of distinctive named entity types (Companies, Products, Bands, Movies and more), and almost all these types are relatively infrequent. On the other hand, tweets often lack sufficient context to determine an entity's type without the aid of background knowledge.

A. Graphs as a Text Representation Structure

A detailed study of different uses of graphs for natural language processing tasks is explained in [11], where it presents algorithmic formulations for:

- Synonyms detection
- Measures of semantic distance on semantic networks
- Textual entailment
- Word-sense disambiguation
- Sentiment classification

In addition to the algorithms and applications covered in [12] several research works have used graphs for representing text data and solving many natural language processing problems.

In the task of document representation Pinto et al. [13] proposed a reliable graph-based representation schema of textual documents that incorporates different levels of formal representation of natural language, and taking into consideration many linguistic levels, such as lexical, morphological, syntactical and semantics and by also extracting useful text patterns in the graph. They state the successful use of their schema in the broader framework of document understanding.

III. PRELIMINARIES AND BACKGROUND

Graphs are a powerful representation of natural language because it is easy to map the syntactic relationships that exist between words or even concepts, thus clearly showing the way they connect with each other.

A. Graph Structure

A graph $G = (V, E)$ is a structure consisting of a set of vertices $V = \{v_i | i = 1, n\}$, some of which are connected through a set of edges $E = \{(v_i, v_j) | v_i, v_j \in V\}$. In a weighted graph $G_w = (V, E, W)$, edges have associated a weight or cost w_{ij} :

$W = \{w_{ij} | w_{ij} \text{ is the weight/cost associated with edge } (v_i, v_j), w_{i,j} \in \mathbb{R}\}$. Edges can be directed or undirected.

Depending on the application for the graphics to be used, the nodes and edges can represent a variety of units and links. Nodes can represent text units as basic as words or as complex as documents. The edges can represent the relationships between these text units, such as: co-occurrence, placement, syntactic structure and lexical similarity [14].

B. Named Entities

NER is an important task in the field of information extraction systems since it aims to locate and classify named entities in raw text into categories previously defined (e.g. Person, Location, Organization). In this way, texts can be represented by their named entities. It is emerged in the Sixth Message Understanding Conference in 1995 [9]. Although, sometimes many of the named entities can be ambiguous to be classified in more than one class, e. g. the automotive company created by Henry Ford in 1903, where "Ford" can be referred to many entities (Name, Company, etc).

On the other hand, NER systems require a large amount of highly accurate training data to perform well at the task named entities recognition [15]. In this way, excellent training data can be achieved by human feedback, since humans can easily differentiate from one context and another, assigning the correct tag to each named entity in the texts.

IV. A GRAPH SCHEMA FOR REPRESENTING NAMED ENTITIES

In this section, the graph schema with enriched language functions is presented and explained. The objective of storing the named entities in a graph is to achieve an expansion in the recognition of named entities through enriched language functions, where synonyms are included as well as semantically similar expressions.

The formal definition of graph proposed to represent these named entities and their relationships, is as follows:

$$G = (V, E, f_E, \alpha),$$

where, V represents a set of vertices or nodes, E are the set of edges that connect to the set of vertices, f_E is the weighting

that the edges receive, and α is the function that calculates the weight that edges receive.

The way in which this structure is built and its operation are explained as follows.

A. Syntax and Structure

The graph is constructed from a finite set of named entities is stored in a .json file. Thanks to the simplicity of this format, it allows great ease of use for many programming languages and also due to its lightness in data storage.

Since the JSON objects are a key-value data format it is convenient to use them to store the information of named entities.

The format used as well as each of the characteristics of the entities that have been employed are illustrated in Figure 1.

```
{
  "level": "1",
  "type": "_INITIAL_",
  "class": "Role",
  "edges": "0.5, 0.1, ",
  "vertices": "4, 6486, ",
  "value": "Digital",
  "id": "3"
},
{
  "level": "2",
  "type": "_Middle_",
  "class": "Role",
  "edges": "0.7, ",
  "vertices": "5, ",
  "value": "Graphic",
  "id": "4"
},
{
  "level": "3",
  "type": "_FINAL_",
  "class": "Role",
  "edges": "",
  "vertices": "",
  "value": "Designer",
  "id": "5"
},
}
```

Fig. 1. Storage format for Named Entities.

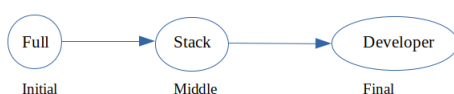


Fig. 2. Possible states of a node in the graph.

Each node is represented with two curly brackets that encapsulate its properties in key-value pairs populating the space between them. These properties are:

- Level: Represents the depth level of the node in the graph.
- Type: Each node in a sequence has a state, as shown in Figure 2. For example, in the sequence: *Full Stack Developer*, the initial node in the sequence is *Full*, after that the intermediate node is *Stack*, and *Developer* is a final node.
- Class: The class refers to the label assigned to that entity of which the node is a part.
- Edges: It contains a list of the weights corresponding to each edge that binds that node with others.
- Vertices: It contains a list of references to the id's of the vertices with which that node is connected.
- Value: It is simply a word, or in other words it is a substring of the named entity of which it is a part.
- Id: The Id or key, is the characteristic through which the node is recognized and needs be unique.

B. The Process of Construction

The set of named entities that are used as Gold Standard is stored in this graph structure, reading one entity at a time until all are completed. The graph begins with a single root node from which branches are added based on the different categories or classes existing for the set of named entities.

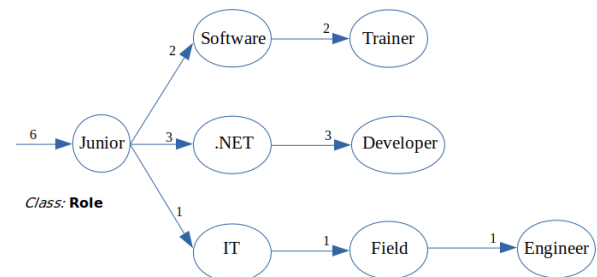


Fig. 3. Coupling of named entities by classes and common shared words to reduce the size of the graph.

In the process of construction of the graph the following steps are taken into consideration:

- Named entities are tokenized.
- For each initial token of the named entity, it is verified if it already exists in the graph with that value and that class. If the entity with that value exists but its class is different then another branch is created from the single root node. If it exists as such, then the weighting of that edge is increased.
- For the intermediate tokens it is verified if that node already exists after the one preceding it. If the node already exists then the weighting of that edge is

increased, if it does not exist, then another edge is created starting from that previous node.

- It may be the case where a named entity exists completely the same and then it only increases the weights, but if there is a variation in one or more tokens then it creates alternate branches. All mentioned before is shown in Figure 3.

V. CASE STUDY: NER

In order to analyze the performance of the structure proposed in this research work, the recognition of named entities comparison is held with other Named Entity Classifiers such as: Stanford NER [16] and CRF++ [2]. The corpus used for the experiment as well as the evaluation measures and methods are explained below.

A. Corpus Used

The corpus was constructed by the collaborative web-based tagger tool for named entities developed in [17]. Since this tool is easy to use and intuitive, allowing to create the necessary classes as well as a fast labeling of documents in plain text. And finally getting the documents labeled in a structured double column format that contains the data prepared to be used in the training of a classifier model provided by Stanford NER [18]. The first column contains the words or tokens of the document and the second column represents the class to which it belongs if it belongs to one, in other cases the value of the second column is 0.

The corpus is made up of job offers in the field of IT, where these documents can be represented by the most relevant concepts it contains. These concepts are grouped into 6 predefined classes:

- *Role*: The position or purpose that someone or something has in a situation, organization, society or relationship.
- *Knowledge*: Understanding of, or information about a subject that someone get by experience or study.
- *Skill*: An ability to do an activity or job well, especially because you have practised it.
- *Character*: The particular combination of qualities in a person or place that makes them different from others.
- *Responsibility*: Something that it is your job or duty to deal with.
- *Talent*: Someone who has a natural ability to be good at something, especially without being taught.

All definitions for classes were taken from the Cambridge Advanced Learner's Dictionary & Thesaurus [19].

B. Data Pre-Processing

To carry out the pre-processing of the information, the same software used to label the documents [17] allows to export them to a structured double-column format, as well as to download the corpus in two data sets: one for training and one for testing (always in a random way, which is perfect for this task.).

It has been implemented considering the principles of *V-fold cross-validation* method. So, only 25% of the whole data set is taken as validation data for testing the model and the remaining 75% is used as training data (corpus). The cross-validation process is then repeated 10 times, each time with random sets of documents (for both the training data set and the test data set) [20].

C. Measures and Training

In order to measure the performance of the NER classifiers, three well known measures are taken into consideration: *Precision*, *Recall* and *F1*. To calculate those measures, first is important to define four main aspects (as shown in Table I).

TABLE I
PARAMETERS TO CALCULATE *Precision*, *Recall* AND *F1*.

Actual Class	Predicted Class		
		Class assigned	No class assigned
		Class assigned	No class assigned
Actual Class	Class assigned	True Positive	False Negative
	No class assigned	False Positive	True Negative

- *True Positive (TP)*: It means that the class predicted by the classifier is the same class that is actually assigned originally.
- *False Negative (FN)*: It is when the word has a class assigned, but the classifier can not predict a class.
- *False Positive (FP)*: It is when the word has not originally assigned a class, but the classifier predicts a class.
- *True Negative (TN)*: It is when the word has not originally assigned a class, and the classifier also fails to assign a class.

Once knowing the essentials of the different metrics, it is possible to define the performance measures in the following way:

- Precision is estimated by $\frac{TP}{TP+FP}$ and represents the correct predicted positives over the total of predicted positives.
- Recall is calculated by $\frac{TP}{TP+FN}$ and it shows how many of the actual positives the model predicted as positives.
- F1 is calculated by $2 * \frac{Precision * Recall}{Precision + Recall}$ and is just the harmonic average of *Precision* and *Recall*.

D. Experiments

For the training of the classifiers, 75% of the documents were randomly obtained with the collaborative web-based tagger tool and the remaining 25% was used to test the classifier, repeating the process 10 times and using exactly the same sets for the 3 classifiers.

The results obtained for the different classes were averaged by execution and classifier due to the large amount of data, and then contrasted according to the three measurements, these results are presented in detail in Table II. From the results

TABLE II
CLASSIFIER RESULTS WITH V-FOLD CROSS VALIDATION METHOD FOR PRECISION, RECALL AND F1.

Execution	Measure	Proposed Graph	Stanford	CRF++
1	Precision	0.25618	0.67076	0.71849
	Recall	0.66611	0.42474	0.22176
	F1	0.37005	0.52012	0.33891
2	Precision	0.24547	0.65652	0.77021
	Recall	0.67437	0.48019	0.28681
	F1	0.35993	0.55468	0.41798
3	Precision	0.26301	0.71343	0.75432
	Recall	0.62482	0.42897	0.22320
	F1	0.37019	0.53579	0.34448
4	Precision	0.23069	0.69565	0.82876
	Recall	0.63487	0.44792	0.23922
	F1	0.33841	0.54495	0.37127
5	Precision	0.24676	0.70243	0.77997
	Recall	0.67871	0.44562	0.23541
	F1	0.36193	0.54530	0.36167
6	Precision	0.25362	0.71091	0.80290
	Recall	0.66431	0.47102	0.26412
	F1	0.36709	0.56662	0.39748
7	Precision	0.24154	0.66375	0.75612
	Recall	0.64164	0.42109	0.22272
	F1	0.35096	0.51528	0.34409
8	Precision	0.20970	0.56629	0.71989
	Recall	0.65850	0.47696	0.25311
	F1	0.31810	0.51780	0.37454
9	Precision	0.23048	0.68721	0.75550
	Recall	0.65269	0.47739	0.25781
	F1	0.34067	0.56340	0.38443
10	Precision	0.24207	0.66593	0.75528
	Recall	0.61821	0.44444	0.24035
	F1	0.34792	0.53309	0.36466

shown in the table it is possible to observe that the best scores for Precision are always achieved by the CRF ++ classifier, leaving far below the graph proposed in each execution.

This means that the number of false positives produced by the CRF ++ classifier is very low, or in other words when the model predicts a class for a named entity, it is correct on average 75% of the time.

On the other hand, as regards the Recall measure, the highest scores are always reached by the proposed graph, showing a great difference in contrast with the CRF ++ classifier. This means that the number of false negatives produced by the proposed graph is low, or what in other words happens is that the proposed graph correctly recognizes on average 64% of the named entities. However, the best results obtained in the F1 measure are always achieved by the Stanford classifier, which means that this classifier has a better balance between Precision and Recall, making it the best performing classifier of the three. For a better understanding of these results, the averages have been captured in charts (from Figures 5 to 7) where it is easy to appreciate the variations and levels reached by each classifier.

The Precision, Recall and F1 measures for the Stanford classifier can be seen in Figure 4 where Precision values are maintained between 65% and 71% in all iterations, with the exception of execution 8, where it drops surprisingly to 56%. This happens because not all documents have the same number of examples for each class. So, in this execution part

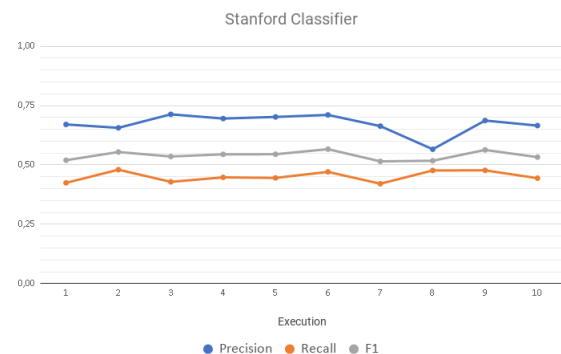


Fig. 4. Average results in all classes for the Stanford classifier.

of the documents that were used for the training provided very few examples of each class in comparison with the other executions. Regarding the measure of Recall, the values are kept constant in an interval ranging from 42% to 48% throughout the experiment. Finally, the F1 measure remains with small variations between 51% and 56% throughout the experiment, showing good performance for the Stanford Classifier.

The second classifier to be compared is CRF++ classifier, and its results for Precision, Recall and F1 measures are showed in Figure 5. Here can be observed that the performance of the Precision measure is very high compared to the Stanford

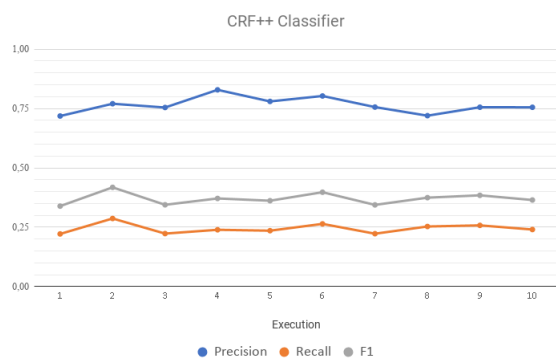


Fig. 5. Average results in all classes for the CRF++ classifier.

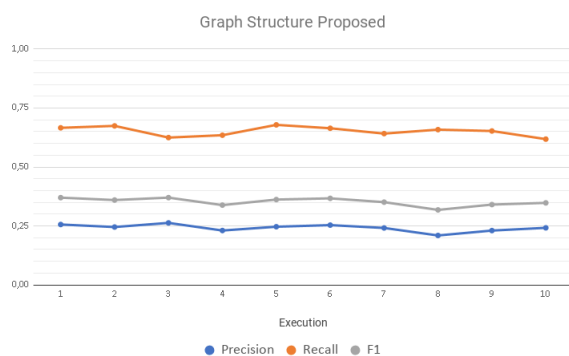


Fig. 6. Average results in all classes for the Graph Structure Proposed.

classifier, being the minimum value reached in iteration 1 with 71% and the maximum value reached in execution 4 with a value of 82%, and the other obtained values oscillate between these two ranges without presenting as major variations. In contrast, the Recall measure is shown too low with a minimum value of 22% in execution 1 and a maximum value achieved in execution 2 with 28%. Presenting even more slight variations in the rest of the executions throughout the experiment. This means that a large number of false negatives is produced by the classifier, or in other words, few truly relevant results were returned. With measures so distant from each other, the F1 measure is maintained in a range that goes from 33% to 41%.

Finally, the results of the graph-based structure proposed are shown in Figure 6. The results are opposite to those of the other classifiers with respect to Precision and Recall since the values appear to be inverted, with a very low Precision but a very high Recall. For the Precision it can be observed that the values remain very similar with a range that goes from 20% to 26%, meaning that, when the classifier predicts a class for a given named entity, it is correct on average 23% of the time. This happens because the proposed model produces a huge number of false positives, so, this model could be improved if a balance of the used classes were achieved.

In the same way for Recall the values remain little variant between 61% and 67%, which shows a more uniform behavior

throughout the experiment in comparison with the Stanford and CRF++ classifiers, this means that the proposed model produces a few false negatives. This occurs thanks to the weights assigned in the graph when more than one named entity is similar in the words that make it up and belong to the same class, so the model clusters the named entities similar and by class and manages to correctly recognize a large proportion entities named in each class. Thus, the harmonic average of these measures represented by F1, can be observed without large drops or sudden peaks throughout the entire experiment, and oscillating between 31% and 37%.

In general, it can be observed that the Stanford classifier performs better in the task of recognizing named entities, maintaining a great balance in Precision and Recall. In second place of performance is the CRF++ classifier with a notorious difference below the performance achieved by Stanford. Finally, the graph structure proposed here is in the last place, but with results not so far from those achieved by the CRF++ classifier. Notwithstanding the above, it is possible to point out that the disk space consumed by the graph proposed here is much smaller than that occupied by the other classifiers, in addition to the training time that is prolonged for the other classifiers. So the proposed graph considerably lighter and faster than the other classifiers.

VI. CONCLUSIONS

In the present research work an enriched graph structure has been proposed to detect named entities. This structure is enriched by using functions and semantic information coming from synonyms. This structure was formally defined taking into account the theory of graphs. In this way, named entities are stored and levels of options or variations are added through synonyms, also using a weighting based on the number of similar entities in the original corpus. On the other hand, the complexity of the graph was reduced by coupling these entities that share common words by category. In addition, the advantage of storing the entities in the graph structure is that it makes it lighter and faster when looking for information. Similarly, this structure allows the possibility of storing more features associated with the semantic relationships between named entities, and that could improve performance in the correct recognition of named entities. Enabling that in a future work the improvement of the results in the metrics used.

Although in developing the experiments and comparing the results obtained from the proposed structure against the Stanford and CRF++ classifiers, it could be observed that the proposed structure has a low performance in terms of Precision but a good performance for the Recall. Meaning that only a small number of positive identifications was actually correct, but that a large proportion of positive positives was correctly identified.

The proposed structure allows great flexibility to store very specific information related to the different named entities, besides using very little disk space as well as less execution

time than the other classifiers and is part of what will be presented in future work.

It is important to emphasize that the structure is in its simple version and that in the next future work it will be enriched language functions by adding more features associated with the semantic relationships between entities as well as the use of a corpus with balanced classes.

REFERENCES

- [1] G. Kowalski, *Information Retrieval Systems: Theory and Implementation*. Boston, MA: Springer US, 2007, pp. 1–23.
- [2] T. Kudo, “CRF++, (version 0.58),” Web, 2005, retrieved from: <https://taku910.github.io/crfpp/download>.
- [3] B. Müller, “Visualization and analysis of extracted information from full text and patent corpora,” Master’s thesis, Fraunhofer Institute for Algorithms and Scientific Computing (SCAI), 08 2009.
- [4] M. A. Hearst, “Untangling text data mining,” in *Proceedings of the 37th Annual Meeting of the Association for Computational Linguistics on Computational Linguistics*, ser. ACL 99. USA: Association for Computational Linguistics, 1999, pp. 3–10. [Online]. Available: <https://doi.org/10.3115/1034678.1034679>
- [5] B. Jansen and S. Rieh, “The seventeen theoretical constructs of information searching and information retrieval,” *Journal of the Association for Information Science and Technology*, vol. 61, no. 8, pp. 1517–1534, 8 2010.
- [6] S. M. Weiss, N. Indurkha, and T. Zhang, *Looking for Information in Documents. In: Fundamentals of Predictive Text Mining*. Springer, 2010.
- [7] M. Allahyari, S. Pouriyeh, M. Assefi, S. Safaei, E. D. Trippe, J. B. Gutierrez, and K. Kochut, “A brief survey of text mining: Classification, clustering and extraction techniques,” 2017.
- [8] D. Nadeau and S. Sekine, “A survey of named entity recognition and classification,” *Linguistic Investigations*, vol. 30, no. 1, pp. 3–26, 2007.
- [18] The Stanford natural language processing group, “NLP. Stanford Named Entity Recognizer,” Web, 4 2019, accessed online <https://nlp.stanford.edu/software/CRF-NER.html>.
- [9] M. Mohamed and M. Oussalah, “Identifying and extracting named entities from wikipedia database using entity infoboxes,” *International Journal of Advanced Computer Science and Applications*, vol. 5, no. 7, 2014.
- [10] A. Ritter, S. Clark, Mausam, and O. Etzioni, “Named entity recognition in tweets: An experimental study,” in *Proceedings of the 2011 Conference on Empirical Methods in Natural Language Processing*. Edinburgh, Scotland, UK.: Association for Computational Linguistics, Jul. 2011, pp. 1524–1534.
- [11] R. Mihalcea and D. Radev, *Graph-Based Natural Language Processing. In: Graph-based Natural Language Processing and Information Retrieval*. Cambridge University Press, 2011.
- [12] —, *Semantics. In: Graph-based Natural Language Processing and Information Retrieval*. Cambridge University Press, 2011.
- [13] D. Pinto, H. Gómez-Adorno, D. Vilariño, and V. K. Singh, “A graph-based multi-level linguistic representation for document understanding,” *Pattern Recogn. Lett.*, vol. 41, no. C, pp. 93–102, May 2014.
- [14] V. Nastase, R. Mihalcea, and D. R. Radev, “A survey of graphs in natural language processing,” *Natural Language Engineering*, vol. 21, no. 5, pp. 665–698, 2015.
- [15] S. Tardif, J. R. Curran, and T. Murphy, “Improved text categorisation for Wikipedia named entities,” in *Proceedings of the Australasian Language Technology Association Workshop 2009*, Sydney, Australia, Dec. 2009, pp. 104–108. [Online]. Available: <https://www.aclweb.org/anthology/U09-1015>
- [16] J. Finkel, T. Grenager, and C. Manning, “Incorporating non-local information into information extraction systems by gibbs sampling,” in *ACL-05, 43rd Annual Meeting of the Association for Computational Linguistics, Proceedings of the Conference*, 01 2005.
- [17] D. Muñoz, F. Pérez-Téllez, and D. Pinto, “Collaborative web-based tagger for named entities in the task of information extraction,” *Pistas Educativas*, vol. 40, pp. 877–893, 12 2018.
- [19] C. Press, *Cambridge Advanced Learner’s Dictionary*. Cambridge University Press, 2008. [Online]. Available: <https://dictionary.cambridge.org/dictionary/english/>
- [20] S. Arlot and A. Celisse, “A survey of cross-validation procedures for model selection,” *Statistics Surveys*, vol. 4, pp. 40–79, 2010.

Dictionary and Corpus-Based Study of Lexical Functions in Spanish

Olga Kolesnikova and Alexander Gelbukh

Abstract—In this work we study semantic and contextual characteristics of four types of verb-noun collocations in Spanish. Each type corresponds to a different lexical function defined in the works of Žolkovskij and Mel'čuk [1, 2, 3] and further elaborated by Apresjan [4, 5]. First, we explain how the typology of lexical functions can be viewed as a consistent way to classify collocations according to their semantic and syntactic patterns. Then, using four lexical functions as well as free word combinations as classes of verb-noun pairs, we examine how they can be identified automatically by supervised learning methods. To build a semantic representation of verb-noun pairs, we used WordNet hypernyms of the verb and the noun. To study contextual properties of the classes, we experimented on a corpus of news. The highest F1-score achieved in the experiments was 0.81 for CausFunc₁ using hypernyms. We found that contextual characteristics were not powerful enough to discriminate among subtle semantic differences of lexical functions: the best F1-score of 0.62 for Real₁ was achieved by GaussianProcessClassifier using raw frequency of context words after removing stopwords from the corpus. Discussing our results, we looked for features which could account for higher or lower results.

Index Terms—Verb-noun collocations, lexical functions, hypernyms, context, supervised learning.

I. INTRODUCTION

EXTRACTING meaning and relations between words has been central to research in computational linguistics and its more technical counterpart, natural language processing (NLP). The majority of techniques are based on statistics obtained from a corpus: words are represented as vectors in a vector space with frequency of context words as vector features. To mine word associations, the distance between vectors is computed: the less the distance, the stronger the relation between the respective words.

Traditionally, word associations are discovered at the paradigmatic and syntagmatic levels. At the paradigmatic level, such associations or lexical relations as synonymy, antonymy, hyponymy/hypernymy, and meronymy/holonymy are defined.

It can be noted that their definitions are semantic-driven, i.e., it is possible to make a meaningful abstraction of associations belonging to the same type and express it in simple terms and patterns, e.g., as in WordNet Reference Manual¹ [6]:

X is a hyponym of Y if X is a (kind of) Y
Y is a hypernym of X if X is a (kind of) Y
X is a meronym of Y if X is a part of Y
Y is a holonym of X if X is a part of Y

Likewise, synonymy and antonymy can be defined²:

X is a synonym of Y if X is the same as Y
X is an antonym of Y if X is the opposite of Y

Concerning syntagmatic word associations, they are more numerous and diverse: the central notion here is syntactic and semantic combinability or compatibility; a word can be characterized or “portrayed” by other words it typically collocates with. Excellent examples of such combinatorial “portraits” are word sketches generated by Sketch Engine, an online corpus-based language processing and lexicographic tool³ [7]. A sketch includes a set of wordlists, where each list is comprised of words with a certain grammatical relation to the query word. For example, if the query word is a noun, then its sketch displays the relations *object_of*, *subject_of*, *modifier*, *modifiers*, and/or, etc. Figure 1 is a partial representation of the sketch for the noun *control* generated on the British Academic Written English Corpus (BAWE)⁴.

Now we will take a closer look at the column of the relation *object_of* containing verbs used with *control* as the direct object forming verbal phrases: *exercise control*, *regain control*, *maintain control*, etc. Reviewing the verbs, it can be noted that, on the one hand, they have different meaning, but on the other hand, they can be grouped in sets according to similar semantics:

{*achieve*, *gain*, *regain*},
{*exercise*, *exert*},
{*maintain*, *retain*}.

Manuscript received on June 12, 2019, accepted for publication on September 20, 2019, published on December 30, 2019.

Olga Kolesnikova is with the Escuela Superior de Cómputo, Instituto Politécnico Nacional, Mexico (e-mail: kolesolga@gmail.com).

Alexander Gelbukh is with the Centro de Investigación en Computación, Instituto Politécnico Nacional, Mexico (web: www.gelbukh.com).

¹ <http://ccl.pku.edu.cn/doubtfire/Semantics/WordNet/Manual.html>

² https://en.wiktionary.org/wiki/Wiktionary:Semantic_relations

³ <https://www.sketchengine.eu>

⁴ BAWE was developed at the Universities of Warwick, Reading and Oxford Brookes under the directorship of Hilary Nesi and Sheena Gardner (formerly of the Centre for Applied Linguistics, Warwick), Paul Thompson (formerly of the Department of Applied Linguistics, Reading) and Paul Wickens (School of Education, Oxford Brookes), with funding from the ESRC (RES-000-23-0800). The corpus includes 2,761 academic works with about 7 million words written at the universities in the UK.

WORD SKETCH				
British Academic Written English Corpus (BAWE)				
control as noun 3,580x ...				
usage patterns	modifier	modifies	object_of	and/or
poss ...	biological ... of biological control	signal ... the control signal	exercise ...	planning ... planning and control
Sfin ...	merger ... merger control "	group ... the control group	regain ...	ownership ... ownership and control
VPto ...	weed ... of weed control	agent ... biological control agents	maintain ... maintain control	power ... power and control
Swh ...	pest ... pest control	treatment ... the control treatment	exert ...	command ...
VPing ...	quality ... quality control	mechanism ... control mechanisms	lose ...	coordination ...
Sing ...	quantum ... adaptive femtosecond quantum control	experiment ... control experiments	match ... matched controls	monitoring ...
	pid- ... than the MZN PID control	loop ... control loop	gain ... to gain control	control ...
	mzn- ... than the MZN PID control	sample ... the control sample	retain ...	reduction ...
	mesh ... mesh control	system ... control system	relinquish ...	management ...
	adaptive ... adaptive femtosecond quantum control	measure ... control measures	strengthen ...	surveillance ...
		valve ...	achieve ...	treatment ...
			assert ...	regulation ...
			▼	▼

Fig. 1. Word sketch of the noun *control*.

{*strengthen, assert*},
{*relinquish, lose*}.

The verbs in each set convey some unique concept which can be possibly formalized as follows:

{*achieve, gain, regain*}: begin_to_carry_out (*control*)
{*exercise, exert*}: carry_out (*control*)
{*maintain, retain*}: continue_to_carry_out (*control*)
{*strengthen, assert*}: carry_out_to_a_greater_extent (*control*)
{*relinquish, lose*}: terminate_to_carry_out (*control*)

In fact, the same semantic concepts can be found in verb-noun relations across different nouns. In Table 1 we present a number of verbal concepts exemplified with the verbs we looked up in the object_of column in sketches for the five nouns: *control*, *support*, *obstacle*, *favour*, and *attention*. The sketches were generated by Sketch Engine on the aforementioned BAWE corpus and the ukWac corpus⁵ (Ferraresi, Zanchetta, Baroni, & Bernardini, 2008). At the beginning of each row, a formalization of the respective semantics is proposed.

The concepts specified in Table 1 and many others alike can be used to characterize and classify the various syntagmatic relations between words thus enabling meaningful generalizations of diverse phrase types. A powerful abstraction

of these semantic concepts is lexical function, a formalism proposed and developed in the works of Žolkovskij and Mel'čuk [1, 2, 3] and further elaborated by Apresjan [4, 5] to represent numerous lexical semantic relations between words in a unified and consistent way.

Lexical function (LF) is defined similarly to a function in mathematics: it is an abstraction of the dependency relation between a word w of a vocabulary V and a set W of words $\{w'_1, \dots, w'_n\}$, $W \subseteq V$. The word w is the argument to the lexical function, and the set W is its value: $LF(w) = W$. Each LF represents a specific lexical semantic relation between the LF argument and each word in the LF value set. About 60 lexical functions have been defined on both the paradigmatic and syntagmatic levels, their detailed descriptions can be found in [9]. Table 2 shows some examples borrowed from [9-13].

This work is a study of four syntagmatic lexical functions most frequently observed in verb-noun collocations. These functions are as follows:

1. Oper₁, from Latin *operari* 'do, carry out', means 'to perform the action given by the noun', e.g. *make a decision, make a step, take a shower, take a walk, commit suicide, do an exercise, give a talk, give a smile, have breakfast, pay a visit, lend support*. The number in the subscript means that the action is realized by the agent, the first argument of the verb

⁵ The ukWac contains texts retrieved by crawling the .uk domain and includes more than 2 billion words.

TABLE I
SEMANTIC CLASSES OF VERBS THAT COLLOCATE WITH NOUNS *CONTROL*, *SUPPORT*, *OBSTACLE*, *FAVOUR*, AND *ATTENTION*.

Noun	<i>control</i>	<i>support</i>	<i>obstacle</i>	<i>favour</i>	<i>attention</i>
Sense⁶	an act or instance of controlling; power or authority to guide or manage	the act or process of supporting, the condition of being supported	something that impedes progress or achievement	friendly regard shown toward another especially by a superior; approving consideration or attention	the act or state of applying the mind to something
cause	<i>enforce, enhance, ensure, establish, impose, offer, provide, set</i>	<i>encourage, ensure, generate, give, lend, offer, provide</i>	<i>cause, create, establish, place, pose, present, provide, raise</i>	<i>bestow, confer, do, give, grant, offer, pay</i>	<i>attract, awaken, bring, captivate, capture, catch, direct, draw, grab, grip, point, pull, trigger</i>
begin_ to_carry_out	<i>achieve, acquire, gain, get, obtain, regain, resume, take</i>	<i>attract, find, gain, get, obtain, receive, win</i>	<i>address, confront, encounter, face, meet</i>	<i>accept, earn, find, gain, get, obtain, receive, win</i>	<i>arrest, center, concentrate, fix, gain, garner, get, give, place, put, turn</i>
carry_out	<i>exercise, exert, have, hold</i>	<i>enjoy, use</i>	<i>deal with, experience, handle, tackle</i>	<i>have, experience, enjoy, use</i>	<i>dedicate, devote, enjoy, exercise, focus, give, occupy, pay, relish</i>
continue_ to_carry_out	<i>continue, develop, ensure, keep, maintain, preserve, retain</i>	<i>continue, maintain</i>	<i>remain</i>	<i>keep</i>	<i>continue, develop, hold, maintain, sustain</i>
make_visible	<i>demonstrate, exhibit</i>	<i>demonstrate, express, reveal, show</i>	<i>identify, show</i>	<i>exhibit, show</i>	<i>display, reflect, show</i>
carry_out_ to_a_greater_ extent	<i>assert, extend, increase, strengthen</i>	<i>extend</i>	<i>increase</i>	<i>show, spread</i>	<i>broaden, extend, force, grow, heighten, increase, widen</i>
carry_out_ to_a_lesser_ extent	<i>decrease, ease, limit, loose, reduce, relax</i>	<i>limit, reduce, remove</i>	<i>minimise, reduce</i>	<i>reduce</i>	<i>avert, confine, decrease, diminish, discourage, divide, limit, minimize, reduce, restrict, shift, split</i>
terminate_ to_carry_out	<i>abolish, lose, relinquish, remove, surrender</i>	<i>end, lose, refuse, withdraw, withhold</i>	<i>avoid, eliminate, fix, ignore, overcome, remove, resolve</i>	<i>lose, withdraw, withhold</i>	<i>deflect, detract, distract, divert, escape, lose, remove, seize, withdraw</i>

- Real₁, from Latin *realis* ‘real’, means ‘to fulfill the requirement imposed by the noun or performing an action typical for the noun’, the action is also carried out by the agent, e.g. *drive a bus, follow advice, spread a sail, prove an accusation, succumb to illness, turn back an obstacle*.
- CausFunc₀ is a complex LF comprised of two semantic units: Caus, from Latin *causare* ‘cause’ and Func₀ from Latin *functionare* ‘function’; CausFunc₀ means ‘to cause the action/event denoted by the noun to happen, occur’, zero in the subscript means that the action is viewed as happening without respect to its agent or that there is no agent, e.g. *bring about the crisis, create/present a difficulty, call elections, establish a system, produce an effect*.
- CausFunc₁ is another complex LF meaning ‘to cause the event of someone performing the action denoted by the

noun’, e.g. *open a perspective, raise hope, open a way, cause damage, instill a habit (into someone)*.

II. MATERIALS AND METHODOLOGY

A. Dataset of Lexical Functions

The objective of this work is to study semantic and contextual properties of the four syntagmatic lexical functions described in the previous section. We intend to examine how these properties would allow for detecting LFs automatically with supervised learning methods. The study was performed on Spanish verb-noun combinations annotated with lexical functions. Table 3 presents a few instances of our dataset. For each LF we borrowed 60 samples from the list of Spanish verb-noun collocations annotated with LFs [14] in order to make our dataset balanced.

⁶ Definitions of senses are borrowed from <https://www.merriam-webster.com/>

TABLE II
EXAMPLES OF LEXICAL FUNCTIONS, ARG IS THE LF ARGUMENT, VALUE IS THE LF VALUE.

Paradigmatic		Syntagmatic	
LF	Definition	LF	Definition
Syn(<i>car</i>) = <i>vehicle</i> Syn(<i>modify</i>) = <i>change</i>	Synonym	Bon(<i>lecture</i>) = <i>informative</i> Bon(<i>meal</i>) = <i>exquisite</i>	From Lat. <i>bonus</i> , good; positive property of Arg
Anti(<i>open</i>) = <i>close</i> Anti(<i>high</i>) = <i>low</i>	Antonym	Degrad(<i>milk</i>) = <i>sour</i> Degrad(<i>tooth</i>) = <i>decay</i>	Degrade, become permanently worse or bad
Conv ₂₁ (<i>give</i>) = <i>receive</i> Conv ₂₁ (<i>include</i>) = <i>belong</i>	From Lat. <i>conversivum</i> , converse; the same action viewed as performed by the agent (Arg) and as performed by the recipient (Value)	Liqu(<i>file</i>) = <i>delete</i> Liqu(<i>law</i>) = <i>annul</i>	Liquidate Arg, cause Arg not to be
Gener(<i>table</i>) = <i>furniture</i> Gener(<i>rose</i>) = <i>flower</i>	Generic concept of Arg	Magn(<i>love</i>) = <i>deep</i> Magn(<i>patience</i>) = <i>infinite</i>	From Lat. <i>magnus</i> , great; intensification of Arg: very, to a high degree, intense, intensely
Sing(<i>fleet</i>) = <i>ship</i> Sing(<i>sand</i>) = <i>grain</i>	A singular instance, unit of Arg	Son(<i>ass</i>) = <i>bray</i> Son(<i>bell</i>) = <i>chime</i>	From Lat. <i>sonare</i> , to sound; typical sound or noise of Arg
Mult(<i>cattle</i>) = <i>herd</i> Mult(<i>bee</i>) = <i>swarm</i>	Multitude of Arg	Manif(<i>amazement</i>) = <i>lurk</i> Manif(<i>joy</i>) = <i>explode</i>	From Lat. <i>manifestare</i> , to manifest; Arg manifests itself (in something)

We achieve our objective by determining the extent to which lexical functions can be automatically identified by supervised learning methods, first, using semantic information obtained from WordNet [15] and, second, using contextual data retrieved from a corpus of 1,131 issues of Excélsior newspaper within the period from April 1996 to June 1999. We explain both methods in two subsections which follow.

B. Semantic Approach

To take advantage of the semantic information provided by WordNet, we extracted all hypernyms of the verb and noun for each verb-noun collocation. As an example consider the collocation *tomar una decisión* (make a decision) annotated with Oper₁.

Hypernyms of *tomar* and *decisión* can be viewed in Table 4 together with sense glosses, below each Spanish synset its corresponding synset in the English WordNet is given, numbers following the word and underscore are sense numbers. The synsets were retrieved from the Multilingual Central Repository version 3.0 [16]. Synsets containing *tomar* and *decisión* were also included as the zero-level hypernyms.

C. Contextual Approach

Contextual data was obtained taking four words to the left of the verb and four words to the right of the noun, words between the verb and the noun were not taken into account in this work. We also studied the impact of stopwords by keeping or removing them from the corpus thus obtaining two context

representations: with and without stopwords. The bag of words model was applied in our experiments, i.e., the word order was disregarded, only word frequencies were considered.

As an example, let us take the same collocation *tomar decisión* considered in section II.B to see its context in the following segment:

Ahora le corresponde el turno a la microeconomía: su gobierno debe **tomar la** decisión sin vacilar en ningún momento de ser factor de unidad por sus acciones determinantes en beneficio de la micro y pequeña empresa (borrowed from the article *Propuesta a Zedillo Sobre la Cartera Vencida* (A proposal to Zedillo about overdue loans), Excélsior, April 6, 1996; literal word-for-word translation: Now to it corresponds the turn to the microeconomics: its government must make the decision without hesitate in no moment to be factor of unity for its actions decisive for benefit of the micro and small business).

The context of *tomar decisión* using the option of keeping stopwords is the set {*microeconomía, su, gobierno, deber, sin, vacilar, en, ningún*}, and the context of this collocation after stopwords elimination becomes {*turno, microeconomía, gobierno, deber, vacilar, momento, factor, unidad*}. Such two types of sets were generated for all occurrences of *tomar decision*, the sets of each type were united to represent the context of this collocation.

TABLE III
EXAMPLES OF SPANISH LEXICAL FUNCTIONS, EACH SPANISH COLLOCATION IS FOLLOWED BY ITS ENGLISH TRANSLATION.

Oper ₁	Real ₁	CausFunc ₀	CausFunc ₁
<i>dar un beso</i> give a kiss <i>ejercer una función</i> exercise a function <i>hacer cálculo</i> do calculation <i>jugar un papel</i> play a role <i>presentar una dificultad</i> present a difficulty <i>realizar una tarea</i> do a task <i>tener sabor</i> have taste (about food)	<i>alcanzar una meta</i> reach a goal <i>aprovechar la oportunidad</i> use the opportunity <i>contestar la pregunta</i> answer the question <i>cumplir el requisito</i> fulfill the requirement <i>lograr el objetivo</i> achieve the objective <i>recorrer un camino</i> walk along a road <i>seguir la instrucción</i> follow the instruction	<i>convocar un concurso</i> call for a contest <i>crear un sistema</i> create a system <i>declarar guerra</i> declare war <i>encontrar el camino</i> find the way <i>establecer un criterio</i> establish a criterion <i>producir un aumento</i> produce an increase <i>provocar un cambio</i> cause a change	<i>dar sentido</i> give sense <i>abrir un espacio</i> open a space <i>ofrecer la oportunidad</i> offer the opportunity <i>prestar ayuda</i> give help <i>reservar el derecho</i> reserve the right <i>poner un límite</i> put a limit <i>hacer realidad</i> make (sth) a reality

TABLE IV
HYPERNYMS OF *TOMAR* AND *DECISIÓN* USED AS FEATURES TO REPRESENT THE MEANING OF THE COLLOCATION *TOMAR UNA DECISIÓN* (MAKE A DECISION). BELOW EACH SPANISH SYNSET, THE CORRESPONDING ENGLISH SYNSET IS GIVEN.

Word	Synset	Synset gloss
<i>tomar</i> (lit. take)	{coger_1 escoger_1 seleccionar_1 elegir_1 triar_1 decantar_1 optar_1 tomar_2} {choose_1 take_10 select_1 pick_out_1}	pick out, select, or choose from a number of alternatives
	{decidir_2 determinar_1 resolver_3 decidirse_1 concluir_4} {decide_1 make_up_one's_mind_1 determine_5}	reach, make, or come to a decision about something
<i>decisión</i> (decision)	{decisión_2 determinación_3 resolución_3} {decision_1 determination_5 conclusion_9}	the act of making up your mind about something
	{elección_2 selección_1} {choice_2 selection_1 option_3 pick_9}	the act of choosing or selecting
	{acción_1 acto_1 hecho_1} {action_1}	something done (usually as opposed to something said)
	{acción_5 acto_2 actividad_humana_1 acción_humana_1} {act_2 deed_2 human_action_1 human_activity_1}	something that people do or cause to happen
	{evento_1 suceso_1} {event_1}	something that happens at a given place and time
	{rasgo_psicológico_1} {psychological_feature_1}	a feature of the mental life of a living organism
	{abstracción_2} {abstraction_6 abstract_entity_1}	a general concept formed by extracting common features from specific examples
	{entidad_1 ente_1} {entity_1}	that which is perceived or known or inferred to have its own distinct existence (living or nonliving)

D. Supervised Learning

To apply the supervised learning methods chosen for our experiments, we represent lists of hypernyms (semantic approach) and context words (contextual approach) of verb-noun pairs as vectors of features in a vector space model. In the semantic approach, binary feature representation was used: 1 signifies that a given hypernym is present among hypernyms of a verb-noun pair, and 0 signifies that it is absent. Within the contextual approach, word counts (raw frequencies) were used as vector features.

We also experimented with another vector representation using tf-idf values for context words as vector features. Binary features in hypernym vectors can also be viewed as numbers, so we computed tf-idf values for these features. Tf-idf is a widely used function to assign weights to words such that the importance of rare words for meaning discrimination is increased, while the influence of very frequent or common words is decreased. Frequent and common words can be found in the context of words with very different semantics; thus, they do not help in distinguishing among different senses, and moreover, they introduce noise into the dataset.

In total, we built six vector representations:

1. Vectors of binary features for hypernoms,
2. Vectors of tf-idf values for hypernoms,
3. Vectors of context word counts on the original corpus,
4. Vectors of context words counts on the corpus after stopwords removal,
5. Vectors of tf-idf values for context words on the original corpus,
6. Vectors of tf-idf values for context words on the corpus after stopwords removal.

We defined the task of automatic identification of lexical functions in verb-noun collocations as a classification task, in which collocations are to be classified into four classes corresponding to the four chosen syntagmatic LFs: Oper₁, Real₂, CausFunc₀, and CausFunc₁.

To the four classes mentioned above, we added free verb-noun combinations as another class to see how they can be detected in contrast to lexical functions. Free word combinations are phrases whose meaning can be derived as a combination of individual word meanings, e.g., *cook a meal*, *give a pen*, *take a box*. On the other hand, the meaning of restricted word combinations or collocations cannot be interpreted using the same compositional approach, e.g., *cook the books*, *give a smile*, *take a bite*.

Concerning supervised learning methods, we selected techniques commonly used in NLP tasks and compatible with our vector representations; we applied them as implemented in the Scikit-learn package for Python with default parameters [17]. In what follows we list the chosen methods, for each method its name in the Scikit-learn implementation is given in parenthesis:

Multinomial Naïve Bayes (MultinomialNB),
 Gaussian Naïve Bayes (GaussianNB)
 Gaussian processes for probabilistic classification (GaussianProcessClassifier),
 K-nearest neighbors vote (KNeighborsClassifier),
 Support vector machine (LinearSVC),
 Decision tree multi-class classification (DecisionTreeClassifier),
 Random forest algorithm (RandomForestClassifier),
 Multi-layer perceptron (MLPClassifier),

In the experiments, 50% of the dataset was used for training, and the other 50% was used for validation. In the next section we present the results of our experiments.

III. RESULTS

In this section, we give the results of classifying verb-noun collocations according to the four syntagmatic lexical functions explained in the introduction and exemplified in section II.A. Free verb-noun combinations were also included as a class. For classification, we used supervised learning methods selected in

section II.D. The results are presented in terms of precision, recall, and F1-score. For classification purposes, precision (P) is defined as the number of true positives (Tp) divided by the sum of the number of true positives and the number of false positives (Fp); recall (R) is the number of true positives divided by the sum of the number of true positives and the number of false negatives (Fn), F1-score (F1) is the harmonic mean of precision and recall:

$$P = \frac{Tp}{Tp+Fp}, R = \frac{Tp}{Tp+Fn}, F1 = \frac{2*P*R}{P+R}.$$

A. Experiments with Semantic Representation

As it was explained in section II.A, hypernoms of the verb and the noun in a verb-noun pair were used as binary (Boolean) features in vectors. A feature in a vector had a value of either 1 (if a hypernym is present among the hypernoms of a verb-noun pair) or 0 (otherwise). In fact, 1 and 0 can be interpreted numerically as counts of the number of times a hypernym occurs in the set of all hypernoms of a verb-noun pair, thus, for each count, tf-idf measure can be computed.

Table 5 displays the results of classifying verb-noun pairs into five classes: four lexical functions (Oper₁, Real₁, CausFunc₀, CausFunc₁) and free verb-noun combinations (FC) with supervised learning methods selected in section II.C. We decided to incorporate verb-noun pairs which are not collocations but free word combinations in order to see how they can be distinguished as opposed to lexical functions.

Table 5 is divided vertically into two sections: the left section entitled as counts gives results for the case where vectors include binary-valued features; the right section of the table entitled as tf-idf contains results for the case where vectors include tf-idf measure calculated for each binary value interpreted numerically. For each classifier and for each class, values of precision (P), recall (R), and F1-score (F1) are given. For convenience, F1-scores are in bold. The best F1-score for each class and for each feature representation is underlined (in other words, it is the best F1-score in each column). The lowest part of the table contains average values of precision, recall, and F1-score for each class. In this row, the best F1-score among all classes and both feature representations is underlined, i.e., it is the highest value among all F1-score values in this row.

Interestingly enough, the best classifiers in this experiment were support vector machine (LinearSVC) and DecisionTreeClassifier. LinearSVC distinguished successfully among lexical functions and free word combinations on binary-valued features and DecisionTreeClassifier showed more efficiency on tf-idf values. Comparing all best F1-score values for classes, it can be noted that the highest value of 0.81 was achieved by LinearSVC on CausFunc₁ using counts as vector features. The best average F1-score was also shown for CausFunc₁ on counts.

TABLE V
RESULTS OF CLASSIFICATION USING HYPERNYMS AS FEATURES.

Classifier	Metrics	Oper ₁	Real ₁	Caus Func ₀	Caus Func ₁	FC	Oper ₁	Real ₁	Caus Func ₀	Caus Func ₁	FC
		counts					tf-idf				
Multinomial NB	P	0.54	0.80	0.42	0.71	0.45	0.45	1.00	0.42	0.90	0.38
	R	0.70	0.33	0.61	0.52	0.61	0.67	0.14	0.55	0.55	0.74
	F1	0.61	0.47	0.50	0.60	0.52	0.54	0.24	0.48	0.68	0.50
Gaussian NB	P	0.57	0.70	0.50	0.48	0.47	0.52	0.61	0.47	0.47	0.40
	R	0.48	0.53	0.45	0.61	0.61	0.48	0.47	0.45	0.45	0.61
	F1	0.52	0.60	0.47	0.53	0.53	0.50	0.53	0.46	0.46	0.48
Gaussian Process Classifier	P	0.44	0.74	0.42	0.70	0.47	0.50	1.00	0.50	0.90	0.31
	R	0.78	0.39	0.35	0.58	0.61	0.70	0.17	0.45	0.55	0.78
	F1	0.56	0.51	0.39	0.63	0.53	0.58	0.29	0.47	0.68	0.44
KNeighbors Classifier	P	0.36	0.60	0.33	0.68	0.44	0.50	0.93	0.50	0.68	0.41
	R	0.67	0.25	0.35	0.52	0.52	0.59	0.39	0.65	0.64	0.57
	F1	0.47	0.35	0.34	0.59	0.48	0.54	0.55	0.56	0.66	0.47
Linear SVC	P	0.59	1.00	0.62	0.96	0.48	0.57	0.84	0.53	0.88	0.43
	R	0.85	0.61	0.68	0.70	0.65	0.74	0.44	0.58	0.67	0.70
	F1	0.70	0.76	0.65	0.81	0.56	0.65	0.58	0.55	0.76	0.53
Decision Tree Classifier	P	0.69	0.75	0.51	0.85	0.52	0.68	1.00	0.56	0.92	0.35
	R	0.74	0.58	0.68	0.70	0.57	0.63	0.58	0.58	0.67	0.74
	F1	0.71	0.66	0.58	0.77	0.54	0.65	0.74	0.57	0.77	0.48
Random Forest Classifier	P	0.59	0.76	0.58	0.84	0.58	0.67	0.83	0.47	0.86	0.39
	R	0.89	0.61	0.68	0.64	0.48	0.59	0.53	0.65	0.58	0.65
	F1	0.71	0.68	0.63	0.72	0.52	0.63	0.64	0.54	0.69	0.49
MLP Classifier	P	0.51	0.79	0.45	0.87	0.40	0.49	0.77	0.44	0.77	0.38
	R	0.67	0.53	0.48	0.61	0.61	0.67	0.47	0.45	0.52	0.61
	F1	0.58	0.63	0.47	0.71	0.48	0.56	0.59	0.44	0.62	0.47
Average	P	0.54	0.76	0.48	0.76	0.48	0.55	0.87	0.49	0.80	0.38
	R	0.72	0.48	0.53	0.61	0.58	0.63	0.40	0.55	0.58	0.68
	F1	0.61	0.58	0.50	0.67	0.52	0.58	0.52	0.51	0.66	0.48

TABLE VI
RESULTS OF CLASSIFICATION USING THE ORIGINAL CONTEXT OF VERB-NOUN PAIRS.

Classifier	Metrics	Oper ₁	Real ₁	Caus Func ₀	Caus Func ₁	FC	Oper ₁	Real ₁	Caus Func ₀	Caus Func ₁	FC
		counts					tf-idf				
Multinomial NB	P	0.18	0.39	0.00	0.75	0.00	0.15	0.00	0.00	0.00	0.00
	R	0.82	0.63	0.00	0.09	0.00	1.00	0.00	0.00	0.00	0.00
	F1	0.30	0.48	0.00	0.16	0.00	0.26	0.00	0.00	0.00	0.00
Gaussian NB	P	0.15	0.38	1.00	0.14	0.00	0.15	0.36	1.00	0.13	0.00
	R	0.73	0.37	0.03	0.06	0.00	0.68	0.37	0.03	0.06	0.00
	F1	0.25	0.38	0.06	0.08	0.00	0.24	0.36	0.06	0.08	0.00
Gaussian Process Classifier	P	0.00	0.18	0.00	0.00	0.00	0.20	0.42	0.00	1.00	0.00
	R	0.00	1.00	0.00	0.00	0.00	0.91	0.70	0.00	0.06	0.00
	F1	0.00	0.31	0.00	0.00	0.00	0.33	0.53	0.00	0.11	0.00
KNeighbors Classifier	P	0.23	0.26	0.27	0.50	0.30	0.23	0.42	0.67	0.36	0.40
	R	0.32	0.33	0.34	0.15	0.27	0.73	0.59	0.06	0.24	0.20
	F1	0.26	0.29	0.30	0.23	0.28	0.35	0.49	0.11	0.29	0.27
Linear SVC	P	0.34	0.37	0.44	0.39	0.28	0.29	0.48	0.53	0.53	0.25
	R	0.50	0.37	0.43	0.26	0.30	0.59	0.81	0.26	0.50	0.07
	F1	0.41	0.37	0.43	0.32	0.29	0.39	0.60	0.35	0.52	0.11

Decision Tree Classifier	P	0.20	0.28	0.24	0.33	0.05	0.26	0.22	0.28	0.33	0.15
	R	0.36	0.30	0.23	0.26	0.03	0.41	0.26	0.29	0.18	0.13
	F1	0.26	0.29	0.24	0.30	0.04	0.32	0.24	0.28	0.23	0.14
Random Forest Classifier	P	0.25	0.23	0.42	0.48	0.30	0.14	0.23	0.32	0.32	0.18
	R	0.36	0.22	0.46	0.35	0.27	0.27	0.22	0.29	0.24	0.13
	F1	0.30	0.23	0.44	0.41	0.28	0.18	0.23	0.30	0.27	0.15
MLP Classifier	P	0.22	0.25	0.39	0.62	0.32	0.17	0.50	0.44	0.55	0.25
	R	0.09	0.67	0.37	0.15	0.27	0.64	0.59	0.11	0.35	0.03
	F1	0.13	0.36	0.38	0.24	0.29	0.27	0.54	0.18	0.43	0.06
Average	P	0.20	0.29	0.34	0.40	0.16	0.20	0.33	0.40	0.40	0.15
	R	0.40	0.49	0.23	0.16	0.14	0.65	0.44	0.13	0.20	0.05
	F1	0.24	0.34	0.23	0.22	0.15	0.29	0.37	0.16	0.24	0.09

TABLE VII
RESULTS OF CLASSIFICATION USING THE CONTEXT OF VERB-NOUN PAIRS AFTER STOPWORDS ELIMINATION

Classifier	Metrics	Oper ₁	Real ₁	Caus Func ₀	Caus Func ₁	FC	Oper ₁	Real ₁	Caus Func ₀	Caus Func ₁	FC
		counts					tf-idf				
Multinomial NB	P	0.00	0.46	0.12	0.00	0.27	0.00	0.00	0.18	0.00	0.00
	R	0.00	0.51	0.29	0.00	0.18	0.00	0.00	1.00	0.00	0.00
	F1	0.00	0.49	0.17	0.00	0.21	0.00	0.00	0.30	0.00	0.00
Gaussian NB	P	0.08	0.38	0.24	0.00	0.00	0.08	0.40	0.21	0.00	0.00
	R	0.25	0.43	0.47	0.00	0.00	0.25	0.51	0.35	0.00	0.00
	F1	0.12	0.41	0.31	0.00	0.00	0.12	0.45	0.26	0.00	0.00
Gaussian Process Classifier	P	0.00	0.45	0.00	0.60	0.00	0.00	0.00	0.14	0.24	0.00
	R	0.00	0.97	0.00	0.45	0.00	0.00	0.00	0.29	0.70	0.00
	F1	0.00	0.62	0.00	0.51	0.00	0.00	0.00	0.19	0.35	0.00
KNeighbors Classifier	P	0.00	0.83	0.12	0.25	0.33	0.00	1.00	0.00	0.20	0.43
	R	0.00	0.14	0.18	0.70	0.12	0.00	0.05	0.00	0.85	0.18
	F1	0.00	0.23	0.14	0.36	0.17	0.00	0.10	0.00	0.32	0.25
Linear SVC	P	0.00	0.57	0.21	0.29	0.50	0.00	0.46	0.20	0.26	0.50
	R	0.00	0.22	0.24	0.60	0.59	0.00	0.16	0.29	0.60	0.29
	F1	0.00	0.31	0.22	0.39	0.54	0.00	0.24	0.24	0.36	0.37
Decision Tree Classifier	P	0.00	0.36	0.21	0.36	0.22	0.00	0.33	0.19	0.55	0.19
	R	0.00	0.11	0.29	0.50	0.35	0.00	0.22	0.24	0.60	0.29
	F1	0.00	0.17	0.24	0.42	0.27	0.00	0.26	0.21	0.57	0.23
Random Forest Classifier	P	0.00	0.56	0.16	0.40	0.11	0.00	0.57	0.23	0.30	0.27
	R	0.00	0.24	0.29	0.60	0.12	0.00	0.22	0.35	0.50	0.35
	F1	0.00	0.34	0.21	0.48	0.11	0.00	0.31	0.28	0.38	0.31
MLP Classifier	P	0.00	0.00	0.23	0.36	0.28	0.00	0.37	0.18	0.27	0.70
	R	0.00	0.00	0.18	0.70	0.71	0.00	0.19	0.18	0.65	0.41
	F1	0.00	0.00	0.20	0.47	0.40	0.00	0.25	0.18	0.38	0.52
Average	P	0.01	0.45	0.16	0.28	0.21	0.01	0.39	0.17	0.23	0.26
	R	0.03	0.33	0.24	0.44	0.26	0.03	0.17	0.34	0.49	0.19
	F1	0.02	0.32	0.19	0.33	0.21	0.02	0.20	0.21	0.30	0.21

B. Experiments with Contextual Representation

In this section, we see how lexical functions and free word combinations can be distinguished by their context. We can also observe the importance of stopwords for distinguishing among classes. Table 6 presents the results of our experiments on the original context of verb-noun pairs, i.e., with stopwords preserved. Table 7 gives the classification results using context after stopwords elimination. The structure and notation of Tables 6 and 7 are the same as those of Table 5, described in section III.A.

First, let us observe the classification results using the original context of verb-noun pairs, i.e., extracting it from the original corpus, without previous stopwords elimination. Concerning the numbers in general, it stands out that they are much lower than those in Table 6, where we used hypernyms as vector features in the experiments. The highest average F1-score in Table 6 is 0.37 for Real₁ using tf-idf for context words, while the highest average F1-score in Table 5 is 0.67 for CausFunc₁ using binary valued features for hypernyms, almost two times bigger. Also, comparing best F1-score values in the

columns, we see that in Table 6 it is 0.60 for Real_1 using tf-idf for context words (the same class and configuration with the best average F1-score), and in Table 5 it is 0.81 for CausFunc_1 on binary valued features for hypernyms. However, it is the same classifier which gave the largest number of best results—LinearSVC—and in both experiments this largest number is the same.

Beside LinearSVC, there were other methods with high results for some classes and vector feature configurations: RandomForestClassifier was efficient on distinguishing CausFunc_0 and CausFunc_1 using counts (0.44 and 0.41, respectively), Multinomial NB showed good results on Real_1 using counts (0.48).

Table 7 presents the results of classification using context words of verb-noun pairs on the corpus after stopwords elimination, so beside our study of the effect of numerical feature representation (counts and tf-idf) on the efficiency of classification, we can observe the importance of stopwords for this task.

On the whole, the numbers in Table 7 are lower than in Table 6, so in comparison with Table 5 where we used hypernyms as vector features, they are very low. Also, there are many zeros in Table 7: some classifiers could not distinguish some classes at all. Oper_1 was hardly distinguished by GaussianNB (F1-score value as low as 0.12), and this value was the same on counts and tf-idf. MLPClassifier could not distinguish Oper_1 using counts and tf-idf as well as Real_1 using counts. However, this classifier achieved 0.25 for Real_1 with tf-idf.

FC were not identified at all by GaussianNB and GaussianProcessClassifier using counts and tf-idf, MultinomialNB distinguished FC with an F1-score of 0.21 on counts, but it was completely unable to distinguish this class with tf-idf. KNeighborsClassifier did not detect CausFunc_0 using tf-idf. MultinomialNB turned out to be most inefficient among all classifiers: it could not detect Oper_1 and CausFunc_1 using counts or tf-idf, also failed to identify Real_1 and FC using tf-idf.

Although in many cases the classes were not detected adequately, the best F1-score among all classes and both feature representations in Table 7 is almost the same as the best F1-score in Table 6: a value of 0.62 was reached by GaussianProcessClassifier on Real_1 using counts. Remember, the best value in Table 6 was 0.60 showed by LinearSVC for Real_1 (the same lexical function!) using tf-idf. Concerning the best average F1-score values, they are also quite close: 0.33 in Table 7 for CausFunc_1 using counts and 0.37 for Real_1 in Table 6 using tf-idf.

IV. RESULTS

In this section we expose some insights we could get analyzing the results given in section III. On the one hand, we focused on semantic and syntactic characteristics of the five classes used in our experiments: four verb-noun lexical functions (Oper_1 , Real_1 , CausFunc_0 , and CausFunc_1) and the class which includes free verb-noun combinations (FC). Our

intention was to find out how such characteristics allow for better or, perhaps, problematic automatic discrimination among the classes by supervised machine learning methods. On the other hand, on the basis of our results, one could also observe how classifiers differed in their performance with respect to the classification task.

While discussing the results, we present them in a concise and graphical form for a more convenient observation. Tables and diagrams in this section will help the reader to take notice of correlations between various properties of lexical functions and features as well as methods found to be most effective for classification. In the first subsection we discuss classifiers' performance with respect to the classes and their feature representation, and in the next subsection we consider the classification results.

A. Classifiers and Feature Representation

This subsection presents a summary of classifiers' efficiency on our classification task. To make further discussion on lexical functions more detailed, we decided to first analyze the performance of each classifier tested experimentally.

Table 8 gives the values of precision and recall averaged over all classes for each feature representation; to compute these values, the numbers in Tables 5-7 we used. However, the F1-score values in Table 8 were not computed as averaged F1-score values borrowed from Tables 5-7, as in such case the mean F1-score value would not represent the relation between the mean values of precision and recall fairly. To give fair F1-score values, we computed them from the precision and recall values in Table 8. The highest F1-score for each classifier is underlined.

Among all classifiers, LinearSVC (support vector machine implemented in the Scikit-learn package, Pedregosa et al., 2011) stands out as it achieved the best F1-score of 0.75 on hypernym counts. The second-best method is RandomForestClassifier with an F1-score of 0.70 also shown on hypernym counts. The lowest F1-score of 0.07 was showed by MultinomialNB on tf-idf values computed for words in the original context of verb-noun pairs. The other technique with the same lowest F1-score was GaussianProcessClassifier tested on the original context word counts.

B. Classification

Figures 2-6 present precision, recall, and F1-score averaged over all classifiers for each class: Oper_1 , Real_1 , CausFunc_0 , CausFunc_1 , and FC. The bar diagrams in the figures show how classification results depend on the feature representation types given by the following numbers: 1 stands for hypernym counts (binary features), 2 stands for hypernym tf-idf values, 3 represents word counts in the original context of verb-noun pairs, 4 denotes the representation comprised of tf-idf values computed for words in the original context of verb-noun pairs, 5 stands for word counts in the context after stopwords elimination, and 6 represents tf-idf for words in the context after stopwords elimination.

TABLE VII
CLASSIFIERS' PERFORMANCE EVALUATION.

Classifier	Metrics	Features					
		Hypernyms		Original context		Context without stopwords	
		counts	tf-idf	counts	tf-idf	counts	tf-idf
Multinomial NB	P	0.62	0.69	0.33	0.04	0.15	0.05
	R	0.54	0.48	0.38	0.25	0.20	0.25
	F1	<u>0.58</u>	<u>0.57</u>	<u>0.35</u>	<u>0.07</u>	<u>0.17</u>	<u>0.33</u>
Gaussian NB	P	0.56	0.52	0.42	0.41	0.18	0.17
	R	0.52	0.46	0.30	0.28	0.29	0.28
	F1	<u>0.54</u>	<u>0.49</u>	<u>0.35</u>	<u>0.33</u>	<u>0.22</u>	<u>0.21</u>
Gaussian Process Classifier	P	0.58	0.72	0.04	0.40	0.26	0.10
	R	0.52	0.47	0.25	0.42	0.36	0.25
	F1	<u>0.55</u>	<u>0.57</u>	<u>0.07</u>	<u>0.41</u>	<u>0.30</u>	<u>0.14</u>
KNeighbors Classifier	P	0.49	0.65	0.32	0.42	0.30	0.30
	R	0.45	0.57	0.28	0.40	0.26	0.22
	F1	<u>0.47</u>	<u>0.61</u>	<u>0.30</u>	<u>0.41</u>	<u>0.28</u>	<u>0.25</u>
Linear SVC	P	0.79	0.70	0.38	0.46	0.27	0.23
	R	0.71	0.61	0.39	0.54	0.27	0.26
	F1	<u>0.75</u>	<u>0.65</u>	<u>0.38</u>	<u>0.50</u>	<u>0.27</u>	<u>0.24</u>
Decision Tree Classifier	P	0.70	0.80	0.26	0.27	0.23	0.27
	R	0.68	0.62	0.29	0.28	0.22	0.27
	F1	<u>0.69</u>	<u>0.70</u>	<u>0.27</u>	<u>0.27</u>	<u>0.22</u>	<u>0.27</u>
Random Forest Classifier	P	0.69	0.71	0.34	0.25	0.28	0.28
	R	0.71	0.59	0.35	0.26	0.28	0.27
	F1	<u>0.70</u>	<u>0.64</u>	<u>0.34</u>	<u>0.25</u>	<u>0.28</u>	<u>0.27</u>
MLP Classifier	P	0.66	0.62	0.37	0.42	0.15	0.21
	R	0.57	0.53	0.32	0.42	0.22	0.26
	F1	<u>0.61</u>	<u>0.57</u>	<u>0.34</u>	<u>0.42</u>	<u>0.18</u>	<u>0.23</u>

Table 9 presents the best values of precision, recall and F1-score (not average values as in Figures 2-6) for each lexical functions and free verb-noun combinations in order to see with what method and feature representation each class was identified best. F1-score values are in bold for convenience. The data in Table 9 can also have a practical application: if a high precision or a high recall is required for a natural language system or tool to function in a more robust manner, the numbers in this table can help a language engineer to choose an adequate method and feature representation.

Among all classes, the best F1-score value in Table 9 is 0.81 (underlined). It was achieved by LinearSVC for CausFunc₁ on hypernym counts as vector features. Actually, it is clear from Table 9 that all best F1-scores were always obtained based on hypernym counts, i.e., binary features, though applying different classifiers. Another interesting observation is that for all lexical functions, context works better in terms of recall with the only exception of free verb-noun combinations for whose detection hypernym information is needed. Concerning precision, higher values were attained by taking advantage of hypernym relations as carriers of semantic information in the case of Oper₁ and Real₁, for the other classes—CausFunc₀,

CausFunc₁, and FC—context worked really well.

According to Table 9, the best classifier in our experiments was LinearSVC (support vector machine); its best F1-score result of 0.81 was demonstrated on CausFunc₁ using hypernym counts. Now let us compare it with the results for the other lexical functions and free verb-noun combinations. The goal is to get some insights into properties of lexical functions which influence the degree of success in their automatic identification.

Table 10 gives the confusion matrix for classification with LinearSVC using hypernym counts. It can be seen there that CausFunc₁ is mostly confused with CausFunc₀, which is in fact not surprising because both of them include a causative semantic element and, consequently, share hypernyms. As an example, consider two CausFunc₁ collocations: *proporcionar un servicio*, *crear un sistema*, and two CausFunc₀ collocations: *ofrecer una posibilidad*, *abrir un espacio*. Their hypernyms are presented in Table 11, where the synsets of the words in these collocations are considered as zero-level hypernyms, English translation is given for each word, common hypernyms are underlined.

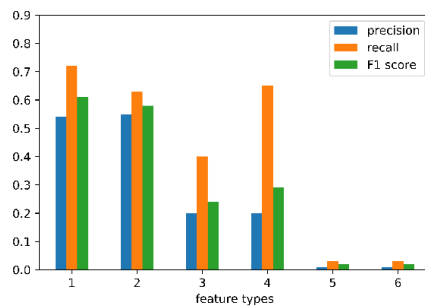
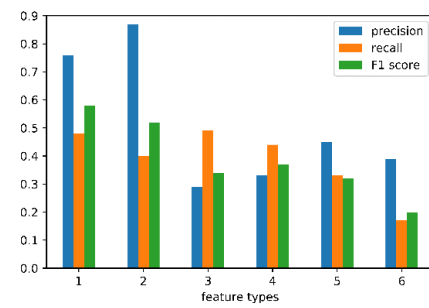
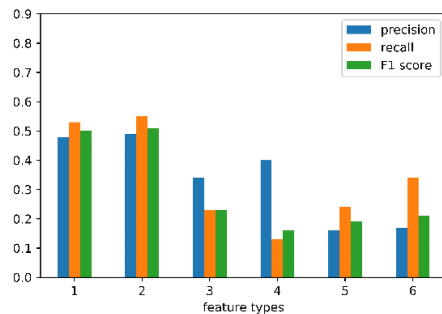
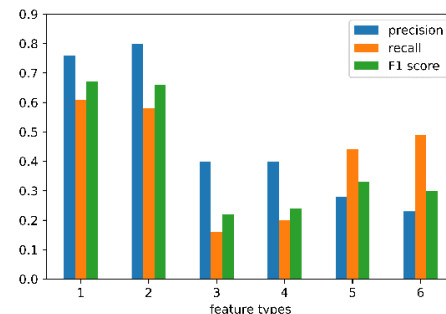
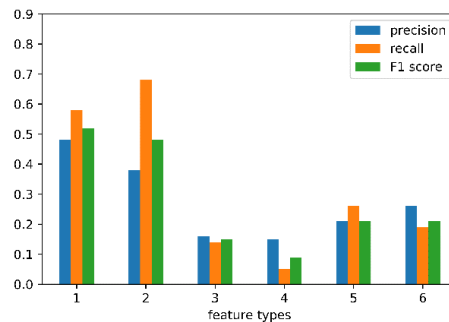
Fig. 2. Oper₁.Fig. 3. Real₁.Fig. 4. CausFunc₀.Fig. 5. CausFunc₁.

Fig. 6. Free verb-noun combinations (FC).

Now let us compare the confusion matrix resulting from LinearSVC with another confusion matrix for the same classifier but on another feature representation: tf-idf for words in the original context, i.e., without stopwords elimination. Let us remark at this point that tf-idf in many cases works better than counts (raw frequency) due to non-uniform frequency of the classes in the corpus, see Table 12. The confusion matrix for LinearSVC referred to previously in this paragraph is displayed in Table 12. For CausFunc₁, this classifier showed the second best F1-score of 0.52, the first best F1-score was 0.60 for Real₁.

It is seen in Table 13 that half of CausFunc₁ samples (17 of 34) were classified as Real₁. In the hypernym representation, no CausFunc₁ pair was classified as Real₁, it was confused not with Real₁ but with CausFunc₀. CausFunc₁ and Real₁ are different in meaning, however, due to this confusion we can suppose that their contexts are similar. Indeed, in the corpus we

used in the experiments, the eight-word window context of 60 CausFunc₁ samples contained 10,449 unique words (we do not consider word frequencies here), the context of Real₁ included 9,705 unique words, and it turned out that both contexts had 5,133 unique words in common.

This high similarity of contexts for two different lexical functions is an interesting detail which does not agree with the distributional hypothesis of word meaning proposed by Harris [18] who assumed that differences in context signal differences in meaning. In fact, this assumption has been widely recognized and applied showing good results in many natural language processing tasks such as topic mining [19], text classification [20], word sense disambiguation [21], sentiment detection [22], authorship attribution [23], among others. However, in our experiments, more subtle semantic differences among lexical functions were not reflected well enough in the context for the classifiers to identify them.

TABLE IX
BEST RESULTS FOR EACH LEXICAL FUNCTION AND FREE VERB-NOUN COMBINATIONS (FC).

LF	Metrics	Value	Feature	Classifier
Oper ₁	P	0.69	hypernyms, counts	DecisionTreeClassifier
	R	1.00	original context, tf-idf	MultinomialNB
	F1	0.71	hypernyms, counts	DecisionTreeClassifier RandomForestClassifier
Real ₁	P	1.00	hypernyms, tf-idf	DecisionTreeClassifier
				MultinomialNB
				GaussianProcessClassifier
			hypernyms, counts	LinearSVC
			context after stopwords elimination, tf-idf	KNeighborsClassifier
	R	1.00	original context, counts	GaussianProcessClassifier
CausFunc ₀	F1	0.76	hypernyms, counts	LinearSVC
	P	1.00	original context, counts	GaussianNB
			original context, tf-idf	
CausFunc ₁	R	1.00	context after stopwords elimination, tf-idf	MultinomialNB
	F1	0.65	hypernyms, counts	LinearSVC
	P	1.00	original context, tf-idf	GaussianProcessClassifier
FC	R	0.85	context after stopwords elimination, tf-idf	KNeighborsClassifier
	F1	0.81	hypernyms, counts	LinearSVC
	P	0.70	context after stopwords elimination, tf-idf	MLPClassifier
FC	R	0.78	hypernyms, tf-idf	GaussianProcessClassifier
	F1	0.56	hypernyms, counts	LinearSVC

TABLE X
CONFUSION MATRIX FOR LINEARSVC ON HYPERNYM COUNTS.

		Predicted class				
		Oper ₁	Real ₁	Caus Func ₀	Caus Func ₁	FC
Known class	Oper ₁	23	0	2	0	2
	Real ₁	7	22	2	0	5
	CausFunc ₀	1	0	21	1	8
	CausFunc ₁	3	0	6	23	1
	FC	5	0	3	0	15

TABLE XI
HYPERNYMS OF VERBS AND NOUNS IN COLLOCATIONS *PROPORCIONAR SERVICIO*, *CREAR SISTEMA*, *OFRECER POSIBILIDAD*, *ABRIR ESPACIO*.

<i>proporcionar</i>	provide	<u>proporcionar</u> , <u>facilitar</u> , <u>surtir</u> , <u>suministrar</u> , <u>dar</u> , <u>transferir</u>	<u>provide</u> , <u>facilitate</u> , <u>supply</u> , <u>deliver</u> , <u>give</u> , <u>transfer</u>
<i>servicio</i>	service	<u>servicio</u> , <u>prestación</u> , <u>trabajo</u> , <u>actividad</u> , <u>acto</u> , <u>acción</u>	service, benefit, work, activity, act, action
<i>crear</i>	create	<u>crear</u> , <u>realizar</u> , <u>causar</u>	create, realize, cause
<i>sistema</i>	system	<u>sistema</u> , <u>método</u> , <u>habilidad</u> , <u>pericia</u> , <u>capacidad</u> , <u>ingenio</u> , <u>poder</u> , <u>cognición</u> , <u>saber</u> , <u>conocimiento</u>	system, method, ability, skill, capacity, ingenuity, power, <u>cognition</u> , <u>knowledge</u> , <u>wisdom</u>
<i>ofrecer</i>	offer	<u>ofrecer</u> , <u>proporcionar</u> , <u>facilitar</u> , <u>surtir</u> , <u>suministrar</u> , <u>dar</u> , <u>transferir</u>	offer, <u>provide</u> , <u>facilitate</u> , <u>supply</u> , <u>deliver</u> , <u>give</u> , <u>transfer</u>
<i>posibilidad</i>	possibility	<u>posibilidad</u> , <u>expectativa</u> , <u>convicción</u> , <u>creencia</u> , <u>contenido mental</u> , <u>cognición</u> , <u>saber</u> , <u>conocimiento</u>	possibility, expectation, conviction, belief, mental content, <u>cognition</u> , <u>knowledge</u> , <u>wisdom</u>
<i>abrir</i>	open	<u>abrir</u> , <u>iniciar</u> , <u>desarrollar</u> , <u>ceder</u> , <u>proporcionar</u> , <u>facilitar</u> , <u>surtir</u> , <u>suministrar</u> , <u>dar</u> , <u>transferir</u>	open, initiate, develop, yield, <u>provide</u> , <u>facilitate</u> , <u>supply</u> , <u>deliver</u> , <u>give</u> , <u>transfer</u>
<i>espacio</i>	space	<u>espacio</u> , <u>área</u> , <u>región</u> , <u>lugar</u> , <u>cosa</u> , <u>objeto inanimado</u> , <u>objeto físico</u> , <u>objeto</u> , <u>entidad</u>	space, area, region, place, thing, inanimate object, physical object, object, entity

TABLE XII
FREQUENCY OF CLASSES IN CORPUS.

Class	Frequency (number of occurrences in corpus)
Oper ₁	63,642
Real ₁	34,250
CausFunc ₀	33,830
CausFunc ₁	46,465
FC	708,159

TABLE XIII
CONFUSION MATRIX FOR LINEARSVC ON TF-IDF OF WORDS
IN THE ORIGINAL CONTEXT

		Predicted class				
		Oper ₁	Real ₁	Caus Func ₀	Caus Func ₁	FC
Known class	Oper ₁	1	2	1	13	5
	Real ₁	1	0	0	4	22
	CausFunc ₀	9	5	4	8	9
	CausFunc ₁	4	17	1	6	6
	FC	2	8	2	14	4

On the other hand, Real₁ was not detected at all by LinearSVC: 22 of 27 Real₁ verb-noun pairs were attributed to free verb-noun combinations. As to their contexts, Real₁ context included 9,705 unique words, FC context included 9,212 unique words, and 4,632 unique words were shared by both contexts. Therefore, due to such contextual lexical similarity of lexical functions and free verb-noun combinations, other feature representations and computational methods are to be looked for in future.

V. CONCLUSION

In this work we studied semantic and contextual characteristics of four syntagmatic lexical functions in Spanish. Our objective was to determine their potential to allow for automatic detection of lexical functions by supervised learning methods. We defined the latter as a classification task.

For experiments, we chose verb-noun collocations of Oper₁, Real₁, CausFunc₀, CausFunc₁, as well as free verb-noun combinations, having a total of five classes. WordNet Hypernyms and context words in a corpus of Spanish news were used as features in a vector space model. The features were represented as their raw frequency and tf-idf values. Also, we studied the impact of stopwords on lexical function detection, so we experimented with the original corpus and the same corpus after stopwords removal.

Concerning supervised learning methods, we chose eight techniques as implemented in the Scikit-learn package for Python. We reported the classification results in terms of precision, recall, and F1-score. The highest F1-score achieved in the experiments was 0.81 for CausFunc₁ using hypernyms. We found that contextual characteristics were not powerful

enough to discriminate among subtle semantic differences of lexical functions: the best F1-score of 0.62 for Real₁ was achieved by GaussianProcessClassifier using context word counts after stopwords removal.

In future, other representations and methods are to be designed in order to attain higher results on the task of lexical function detection.

ACKNOWLEDGEMENT

The work was done under partial support of Mexican Government: SNI, COFAA-IPN, BEIFIIPN, CONACYT grant A1-S-47854, and SIP-IPN grants 20196021, 20196437.

REFERENCES

- [1] A. Zolkovskij and I. Mel'čuk, "On a possible method and tools for semantic synthesis" (in Russian), *Naučno-Tekničeskaja Informacija*, vol. 5, pp. 23–28, 1965.
- [2] A. Zolkovskij and I. Mel'čuk, "On a system for semantic synthesis. I. Structure of the dictionary" (in Russian), *Naučno-Tekničeskaja Informacija*, vol. 11, pp. 48–55, 1966.
- [3] A. Zolkovskij and I. Mel'čuk, "On semantic synthesis" (in Russian), *Problemy kibernetiki*, vol. 19, pp. 177–238, 1967. Translated into French as: A. Zolkovskij and I. Mel'chuk, "Sur la synthèse sémantique", *T.A. Informations*, vol. 2, pp. 1–85, 1970.
- [4] J. D. Apresjan, "On semantic motivation of lexical functions in collocations" (in Russian), *Voprosy Jazykoznanija*, vol. 5, pp. 3–33, 2008.
- [5] J. D. Apresjan, "The theory of lexical functions: An update", in *Proc. Fourth Int. Conf. on Meaning-Text Theory*. Montréal: OLST, 2009, pp. 1–14.
- [6] C. Fellbaum, Ed., "WordNet: An electronic lexical database". Cambridge, MA: MIT Press, 1998.
- [7] A. Kilgarriff and D. Tugwell, "Sketching words", in *Lexicography and Natural Language Processing: A Festschrift in Honour of B. T. S. Atkins*, M.-H. Corréard, Ed. Euralex, 2002, pp. 125–137.
- [8] A. Ferraresi, E. Zanchetta, M. Baroni and S. Bernardini, "Introducing and evaluating ukWaC, a very large web-derived corpus of English", in *Proc. 4th Web as Corpus Workshop (WAC-4) Can We Beat Google*, 2008, pp. 47–54.
- [9] I. A. Mel'čuk, "Lexical functions: A tool for the description of lexical relations in a lexicon", in *Lexical Functions in Lexicography and Natural Language Processing*, L. Wanner, Ed. Amsterdam, Philadelphia, PA: Benjamins Academic Publishers, 1996, pp. 37–102.
- [10] I. A. Mel'čuk, "Collocations and lexical functions", in *Phraseology. Theory, Analysis, and Applications*, A. P. Cowie, Ed. Oxford: Clarendon Press, 1998, pp. 25–53.
- [11] D. Heylen, K. G. Maxwell and M. Verhagen, "Lexical functions and machine translation", in *Proc. 15th Conf. on Computational Linguistics*, Kyoto, 1994, pp. 1240–1244.
- [12] T. Fontenelle, "Ergativity, collocations and lexical functions", in *Proc. EUROLEX*, M. Gellerstam et al., Eds, 1996, pp. 209–222.
- [13] S. H. Song, "Zur Korrespondenz der NV-Kollokationen im Deutschen und Koreanischen", *언어학*, vol. 44, pp. 37–57, 2006.
- [14] A. Gelbukh and O. Kolesnikova, "Supervised learning for semantic classification of Spanish collocations", in: *Advances in Pattern Recognition*, Springer Berlin Heidelberg, 2010, pp. 362–371. The lexical resource (list of Spanish lexical functions) can be downloaded at <http://www.gelbukh.com/lexical-functions/>
- [15] G. A. Miller, C. Leacock, R. Tengi and R. T. Bunker, "A semantic concordance", in *Proc. Workshop on Human Language Technology*, Association for Computational Linguistics, 1993, pp. 303–308.
- [16] A. Gonzalez-Agirre, E. Laparra, G. Rigau and B. C. Donostia, "Multilingual central repository version 3.0: Upgrading a very large

- lexical knowledge base”, in *Proc. GWC 2012 6th International Global Wordnet Conference*, 2012, pp. 118–125.
- [17] F. Pedregosa, G. Varoquaux, A. Gramfort, V. Michel, B. Thirion, O. Grisel, M. Blondel, P. Prettenhofer, R. Weiss, V. Dubourg and J. Vanderplas, “Scikit-learn: Machine learning in Python”, *J. of Machine Learning Research*, vol. 12, pp. 2825–2830, 2011.
- [18] Z. S. Harris, “Distributional structure”, *Word*, vol. 10, no. 23, pp. 146–162, 1954.
- [19] C. Huang, Q. Wang, D. Yang and F. Xu, “Topic mining of tourist attractions based on a seasonal context aware LDA model”, *Intelligent Data Analysis*, vol. 22, no. 2, pp. 383–405, 2018.
- [20] G. Feng, S. Li, T. Sun and B. Zhang, “A probabilistic model derived term weighting scheme for text classification”, *Pattern Recognition Letters*, vol. 110, pp. 23–29, 2018.
- [21] D. Ustalov, D. Teslenko, A. Panchenko, M. Chernoskutov, C. Biemann and S. P. Ponzetto, “An unsupervised word sense disambiguation system for under-resourced languages”, *arXiv preprint arXiv:1804.10686*, 2018.
- [22] H. Saif, Y. He, M. Fernandez and H. Alani, “Contextual semantics for sentiment analysis of Twitter”, *Information Processing & Management*, vol. 52, no. 1, pp. 5–19, 2016.
- [23] B. Alhijawi, S. Hriez and A. Awajan, “Text-based Authorship Identification-A survey”, in *Proc. Fifth Int. Symposium on Innovation in Information and Communication Technology*, IEEE, 2018, pp. 1–7.
- [24] K. A. Overmann, and L. Malafouris, “Situated cognition,” in *The International Encyclopedia of Anthropology*, 2018, pp. 1–8.
- [25] F. Engelmann, S. Granlund, J. Kolak, M. Szreder, B. Ambridge, J. Pine, A. Theakston, and E. Lieven, “How the input shapes the acquisition of verb morphology: Elicited production and computational modelling in two highly inflected languages,” *Cognitive Psychology*, vol. 110, pp. 30–69, 2019.
- [26] R. H. Baayen, Y. Y. Chuang, E. Shafaei-Bajestan, and J. P. Blevins, “The discriminative lexicon: A unified computational model for the lexicon and lexical processing in comprehension and production grounded not in (de) composition but in linear discriminative learning,” *Complexity*, 2019.

Word Embeddings and Length Normalization for Document Ranking

Sannikumar Patel, Markus Hofmann, and Kunjan Patel

Abstract—Distributed word representation techniques have been effectively integrated into the Information Retrieval retrieval task. The most basic approach to this is mapping a document and the query words into the vector space and calculating the semantic similarity between them. However, this has a bias problem towards documents with different lengths, which rank a small document higher compared to documents with larger vocabulary size. While averaging a document by mapping it into vector space, it allows each word to contribute equally, which results in increased distance between a query and the document vectors. In this paper, we propose that document length normalization should be applied to address the length bias problem while using embedding based ranking. Therefore, we have presented an experiment with traditional Length Normalization techniques over, *word2vec* (Skip-gram) model trained using the TREC Blog06 dataset for ad-hoc retrieval tasks. We have also attempted to include relevance signals introducing a simple Linear Ranking (LR) function, which considers the presence of query words in a document as evidence of relevancy while ranking. Our combined method of Length Normalization and LR significantly increases the Mean Average Precision up to 47% over a simple embeddings based baseline.

Index Terms—Word embeddings, neural information retrieval, distributed word representations, Word2Vec.

I. INTRODUCTION

MATCHING semantically similar documents to a query is a core challenge in ad-hoc retrieval. For large scale search engines, this problem does not seem too complicated, as it is possible to identify similar documents by considering user behavioral data such as clicks and hyperlinks as ranking measures [1]. However, for many other Information Retrieval (IR) tasks, it is quite challenging to identify a relevant set of documents and rank them correctly. Current methods for relevancy matching are mainly based on various term-frequency based approaches. However, these methods fail to correctly identify relevant matches whenever a document contains the query words without being relevant, or it is using different vocabulary. These existing methods for similarity matching measure a count of query words in the set of documents.

BM25 is such a traditional model that considers query count as evidence of similarity in the document [2]. The main idea

behind this model is to consider a count of query words in a document as evidence of similarity, whereas non-query words are less useful for ranking. However, this model is less useful when looking for similarity in semantically different documents.

Addressing this problem requires a set of techniques that consider word semantics at the root and not solely depend on term-frequency. One such semantically diverse method is Distributed Word Representation, which captures a precise semantic word relationship by learning high quality distributed vector representations [3]. Distributed Word Representations are based on the idea of statistical language modeling and feature learning where words or phrases from the vocabulary are mapped to vectors of real numbers [4], [3]. A Skip-gram and Continuous Bag of Word (CBOW) are examples of such a method. These methods use Neural Network models to learn vector representation of words from unstructured text data [4]. The main objective of these models is to find word representations that are useful for predicting the surrounding words in a sentence or context. Vectors generated by this model are useful when considering the semantic similarity between words or documents. For this reason, the Neural Network-based Distributed Word Representation approaches are widely used in Natural Language Processing (NLP) tasks such as automatic question answering and machine translation [5], [6]. However, after observing its success and usefulness in a wide range of NLP tasks, some studies have started utilizing it for query-document similarity matching in ad-hoc retrieval [7], [8], [9], [10], [11]. The core idea is to map all document terms against query terms in a shared semantic space and formulating a similarity score by applying measuring techniques such as *cosine similarity* or *euclidean distance*. Therein it gives a relevancy score between query and document, which later can be used to rank documents [8].

In document ranking, we wish to reasonably rank documents without biasing them based on different factors such as its length, vocabulary size, the occurrence of query word, *etc.* In a traditional term-frequency based ranking systems, there are plenty of ways for rewarding and penalizing documents based on such specific parameters. One such inevitable approach is the normalization of the document's length. The length of target documents is one of the most significant factors which considerably affects its ranking [12], because the same term may repeatedly occur in long documents. For example, in a document about a *dog* that has 1,000 terms, it is more likely to have a frequent occurrence of *dog*, than a document with just

Manuscript received on April 27, 2019, accepted for publication on August 12, 2019, published on December 30, 2019.

The authors are with the Technological University Dublin, Ireland (e-mail: Patel.Sannikumar@mytudublin.ie, Markus.Hofmann@tudublin.ie, kunjanpatel@hotmail.com).

300 terms. This will help documents with increased length to being ranked first even if it is not necessarily relevant to the query.

In the contrary, higher ranking of long documents in term-frequency based approaches, are usually ranked lower in embeddings based methods, because with a higher number of terms in the document its centroid is also likely to be sparse (*as every word in document contributes equally*) which results in increased distance between query vector and centroid. The proof is shown in Fig. 1a where long documents are ranked lower despite being relevant (*red * in the figure*) to the query. However, after normalization, the size of the document gets decreased as additional noisy terms are removed, and as a result, relevant documents are ranked higher.

Fig. 1b shows how relevant documents are pushed forward after length-normalization. The main aim of this study is to evaluate whether it is possible to reduced length biased ranking in the embedding approach by applying traditional document normalization techniques. Therefore, we present an experiment with two normalization techniques over the Blog06 dataset [13]. Moreover, for evaluation purpose, we have used Mean Average Precision (MAP). In subsequent sections, we discuss the methodology and a set of experiments performed over the embeddings space model and Length Normalization methods. Then we discussed the results, followed by our conclusion.

II. RELATED WORK

Word embeddings learned unsupervisedly have been surprisingly successful in many NLP tasks. In fact, in many NLP architectures, they have almost completely replaced traditional distributed feature techniques such as Latent Semantic Analysis (LSA) [14]. For many years vector space models have been used in word semantics related tasks and evolved over-time. Latent Dirichlet Allocation (LDA) and LSA are example of models used extensively before the invention of neural word embeddings [15], [14].

A foundation for neural word embeddings were established after the introduction of a unified architecture for NLP [16]. However, term *word embeddings* appeared first in 2003 when researchers presented a study training joint neural language models [17]. The Neural Networks based Skip-gram and CBOW model introduced in 2013 made word embeddings popular among the NLP community, and a toolkit of this model called *word2vec* was made available [4]. Despite being so popular, one major downside of *word2vec* is dealing with terms missing in embedding. To address this problem, an extension to this called *FastText* was introduced. In contrast to *word2vec*, *FastText* treats each word as a composition of character n-grams [18]. Further, researchers released the co-occurrence matrix-based model, signaling that word embedding has reached the mainstream in NLP [19]. After this, many advanced embeddings models have been invented using Neural Networks, overcoming different problems such as missing words and the context of words [20], [21].

However, the earliest study with word embedding for IR was anticipated for retrieval and clustering experiments in which the traditional topic model was used to calculate word embeddings [22]. Usually, strategy for using word embeddings in IR involves deriving a dense vector representation for the query and document terms from embedding space, and then an average of the derived vectors is used as document vector to measure semantic similarity between them, which is relatively easy and popular in the community [7], [8]. After an introduction of word embeddings for IR tasks, it has grown beyond what can be concisely described here. Especially, *word2vec* saw wider adoption in the community [23], [24], [25], [26]. Further, in another study, the impact of word embeddings on scoring in practical IR has evaluated extensively by re-weighting terms using Term-Frequency scores and re-ranking documents based on Word Mover Distance (WMD) [27], [28]. In an attempt to ad-hoc retrieval, studies has also presented experiments with multiple vector spaces (*known as IN and OUT*) available in *word2vec* [4], [25].

In which, they tried to map a query and document terms in different vector spaces (*such as query into IN and document into OUT*) and then ranked them based on cosine similarity scores [25].

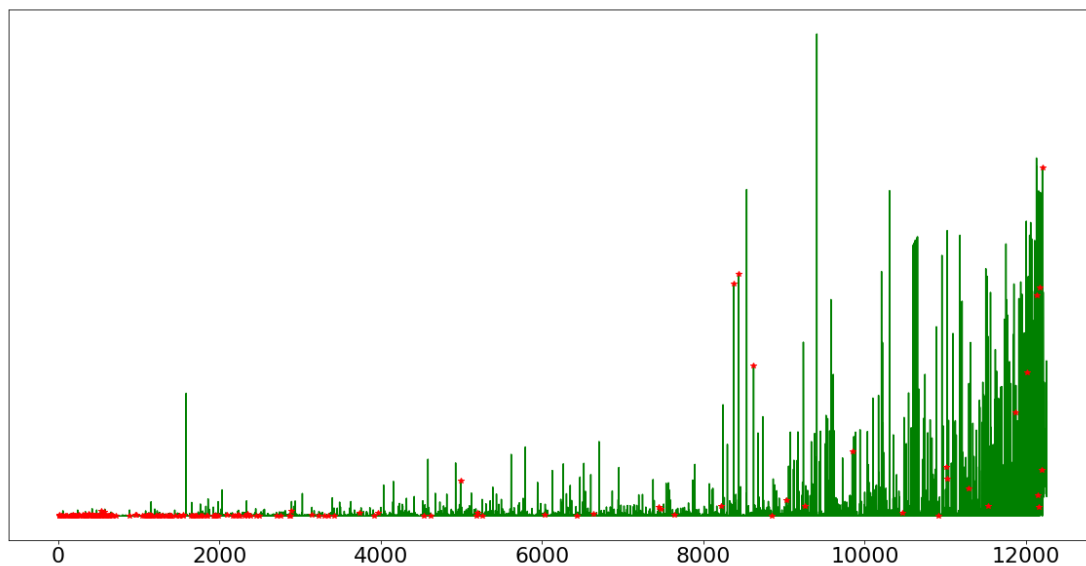
However, word embeddings are not just used for similarity matching between query and document some studies have demonstrated that word embeddings such as *word2vec* and GloVe, trained globally, can be used for query expansion [29]. In another similar study, the author has introduced an Artificial Neural Network classifier to predict the usefulness of expanded query terms over word embeddings. They concluded that terms selected by the classifier for expansion significantly improve retrieval performance [30]. Further, instead of just averaging word vectors, a generalized language model has constructed to derive transformation probabilities between words by using word embeddings, and it showed significant improvement over language model baselines [24]. Beyond this, few studies have also explored learning embeddings by utilizing clickthrough and session data [31], [32]. We can say concisely that, with the new development in Neural Networks, word embeddings are further improving, opening new possibilities for its application to IR tasks.

III. METHODOLOGY

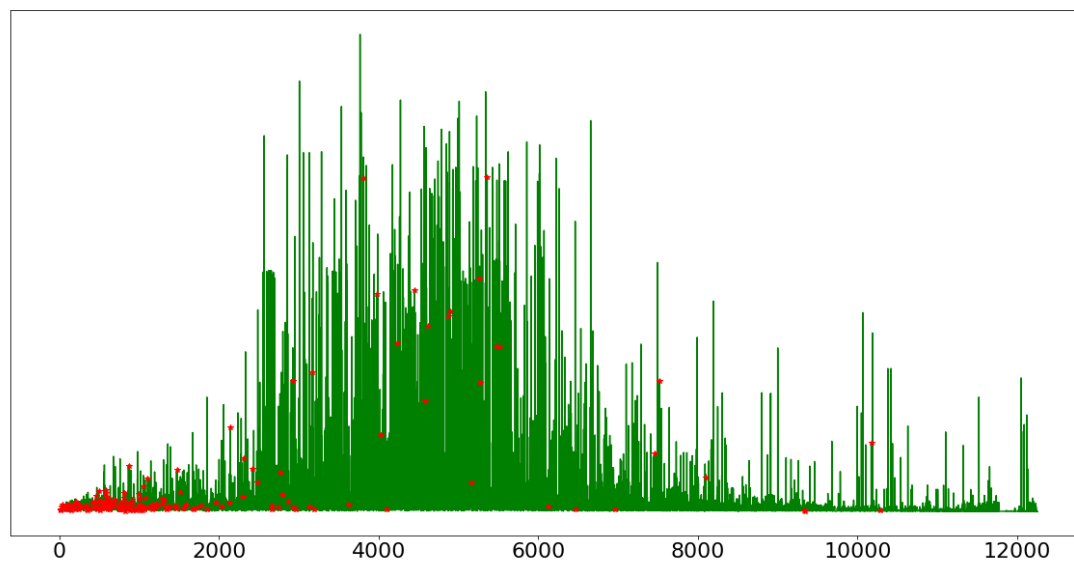
In this section, we first give a formal introduction to the distributed word representation techniques and Document Length Normalization. Then, we present our embedding space model using Length Normalization and Linear Ranking for document ranking.

A. Distributed Word Representation

Many traditional frequency-based word representation techniques such as Term Frequency - Inverse Document Frequency, and co-occurrence matrix, are less efficient to capture word semantics. Therefore, its applicability is less



(a) Large documents are ranked lower in simple embedding based approach.



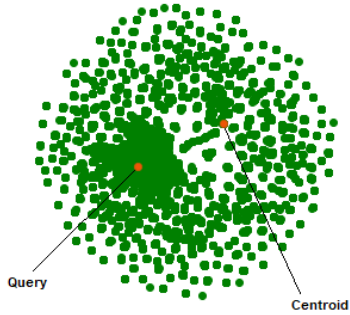
(b) Ranking improves with length-normalization as large documents are pushed forward.

Fig. 1. Projection of initial 12K ranked documents along with their length before normalization and after normalization while using the embedding space model for ranking. (red [*] denotes a relevant document to the selected query).

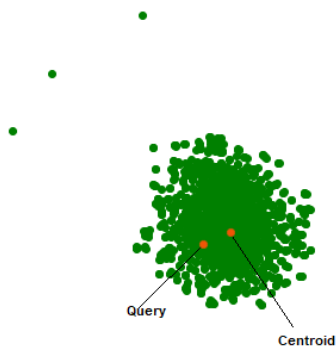
effective for document ranking as it does not preserve semantic relations, resulting in a loss of useful information. However, to deal with this problem, the researchers have started using distributed representation based CBOW and Skip-gram techniques, which preserves word semantics. So far, these techniques have outperformed in various NLP tasks.

The widely used Skip-gram and CBOW are the models in this category that learn word embeddings based on the probability concept. Such as predicting the occurrence of a word around other words in a fixed-length window. In CBOW, the word in a window serves as an input and then

the model attempts to predict context words. Opposite to this, a Skip-gram model attempts to achieve the reverse of what the CBOW model does by predicting a context from a given set of target words [4]. Even though CBOW and Skip-gram use different formulation techniques, they are simple Neural Networks with a single hidden layer to perform a particular prediction task based on sentence semantics. Weights learned in this hidden layer are called vectors, and each vector in the layer is associated with a single word representing the context of the word, which is also called *embeddings*. In general, word embeddings are used in a wide range of NLP tasks such as



(a) Without length-normalization, query and centroid are mapped far from each other in vector space.



(b) With length-normalization, query and centroid get closer.

Fig. 2. The two dimensional PCA projection of embeddings for the document relevant to the query. The query and a centroid are labeled to show the difference in distance between them, prior and after the Length Normalization.

binary classification, question answering, machine translation, *etc.* However, in this paper, we use it for query-document similarity matching by using its unique property of formulating document meaning.

For all experiments in this paper, we have used vectors generated by the Skip-gram model as embeddings to measure the similarity between query and document. However, the proposed ranking model can work with the vectors generated by the CBOW method as well. A comprehensive introduction to CBOW and Skip-gram models is outside the scope of this study, but more details can be found in [4].

B. Document Length Normalization

As discussed in the previous section, the length of the document influences its ranking, as it allows each word in the document to contribute equally rather than their actual relevancy to the query. Therefore, this leads to an unreasonably higher ranking of short documents and lower-ranking of important documents. The large documents are profoundly

affected as with extended vocabulary, it generates centroids, which mapped far from a query in the actual vector space. Fig. 2a shows how the query word and centroid is mapped on distance, compared to Fig. 2b where distance is reduced after normalization.

Therefore, to counter this problem, we normalize the length of documents before calculating a centroid as it removes unnecessary terms, which help in reducing bias among documents with different lengths.

We have applied two different normalization techniques discussed in subsequent sections.

1) *Cosine Normalization*: Cosine Normalization is a widely used method in the vector space model [12]. It addresses the two main problems, higher tf_s and the number of terms in the document at a time. With increased individual term frequency for w_i it increases its $tf * idf$ value, which increases the penalty on the term weights. As can be seen in Equation 1, the Cosine normalization is calculated using square root of each term frequency where, w_i is the $tf * idf$ weight for a term i :

$$\sqrt{w_1^2 + w_2^2 + w_3^2 + \dots + w_i^2}. \quad (1)$$

Also, if the document has more terms, the number of individual weights in the Cosine factor (t in the above formula) increases, yielding a higher normalization factor.

When classifying documents into various categories, Cosine normalization tends to favor retrieval of short documents but suppresses the long documents [12]. So, to address this problem, Pivoted Normalization is suggested.

2) *Pivoted Normalization*: The Pivoted normalization scheme is based on the principle that the probability of retrieval of a document is inversely related to the normalization factor used in the term weight estimation for that document. The higher the value of the normalization factor for a document, the lower chance of retrieval for that document. This relationship suggests that to boost the chances of retrieval for documents of a certain length, we should lower the value of the normalization factor for those documents, and vice-versa.

Figure 3 illustrates the basic idea of Pivoted normalization. In which, the point where the retrieval and relevance curves cross each other is called the pivot. The documents on one side of the pivot are generally retrieved with a higher probability than their relevance probability, and the documents on the other side of the pivot are retrieved with a lower probability than their probability of relevance. A more detail explanation can be found in [12]. However, here we include an explanation of the formula for Pivoted Normalization p :

$$p = (1.0 - slope) * pivot + slope * old_norma, \quad (2)$$

where, old_norma is a normalized vector to its unit length and parameters $slope$ and $pivot$ are selected after cross-validation.

While using pivoted normalization, the new term weight Tw for each term T in the document can be written as:

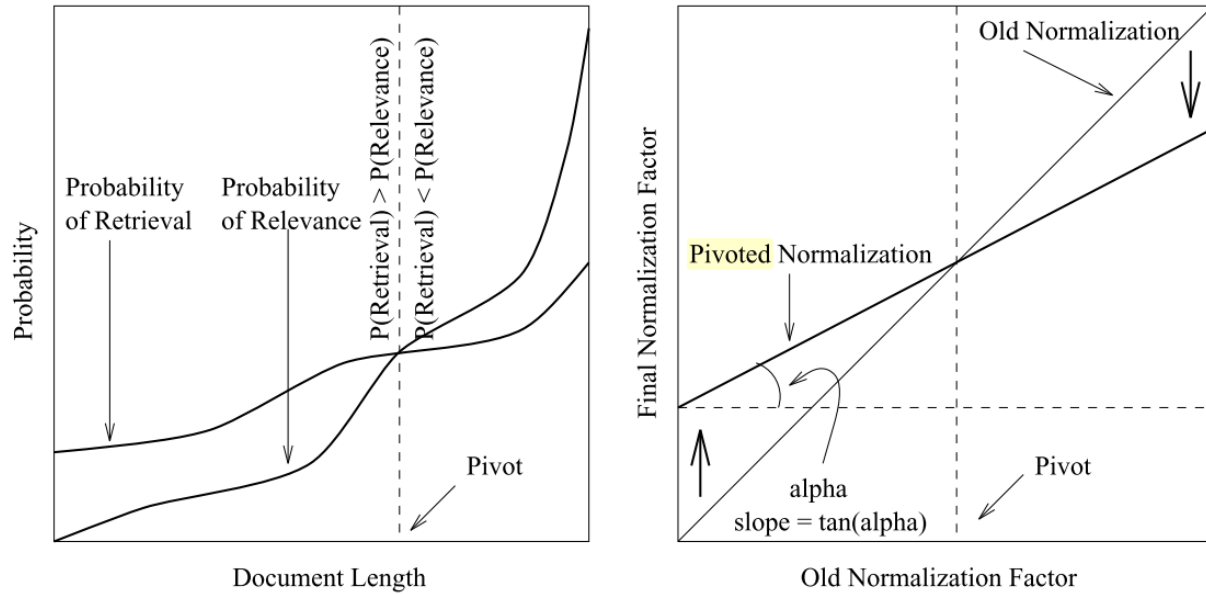


Fig. 3. Pivoted Normalization: The normalization factor for documents for which $P(\text{retrieval}) > P(\text{relevance})$ is increased, whereas the normalization factor for documents for which $P(\text{retrieval}) < P(\text{relevance})$ is decreased [12].

$$T_w = \frac{tf.idf}{(1.0 - slope) * pivot + slope * old_norma}. \quad (3)$$

Both normalization techniques discussed above generate a term-frequency score, which signifies the importance of the word in a document. We use this score as a parameter that serves as a threshold, so any word with the value less than a threshold (min_x) is removed from the document before calculating a centroid. In our study, the optimum value for min_x is selected after parameter switching between values 0.01 to 0.1.

In Fig. 2, it can be seen that the cosine distance between a query and the centroid is shorter for normalized document. We are not discussing both normalization techniques comprehensively in this study, but more relevant information can be found in [33], [12].

C. Embedding Space Model

As discussed earlier, differentiating whether a document merely contains a query word or is genuinely relevant to the query topic is a significant challenge. We attempt to address this problem by utilizing word embeddings with a similar approach to [25], wherein a centroid is calculated first for a document, and cosine distance between the query vector and centroid is considered as a similarity measure. However, we also cannot ignore the fact that the presence of a query word in the document is also a piece of strong evidence for relevancy, and embedding space models are weak rankers in the long run [25]. We define a simple ranking function that

takes query presence into account while ranking:

$$sim(Q, D) = \bar{L} + cos(\vec{Q}, \vec{D}), \quad (4)$$

where:

$$\bar{L} = \sum_{q_i \in Q} \frac{cos(\vec{Q}, \vec{D}) \cdot |q^i| \epsilon D}{|D|}, \quad (5)$$

where:

$$cos(\vec{Q}, \vec{D}) = \frac{1}{|Q|} \sum_{q_i \in Q} \frac{q_i^T \vec{D}}{\|q_i\| \|\vec{D}\|}, \quad (6)$$

where:

$$\vec{D} = \frac{1}{|D|} \sum_{d_i \in \bar{N}(D)} \frac{\vec{d}_i}{\|\vec{d}_i\|}, \quad (if \ d_i > min_x), \quad (7)$$

where, $\bar{N}(D)$ is a length-normalization for document D and min_x is a threshold value.

Here, $cos(\vec{Q}, \vec{D})$ is an absolute relevancy score between query Q and document D calculated using cosine similarity across each query term q_i and centroid \vec{D} . The value of \vec{D} is the mean of all word vectors in document D , which serves as single embedding. A function \bar{N} is applied on document D before calculating centroid \vec{D} . The \bar{N} is a length-normalization function such as Pivoted or Cosine.

\bar{L} serves as linear ranker which considers a count of query term q^t in document D and multiplies it with the cosine score

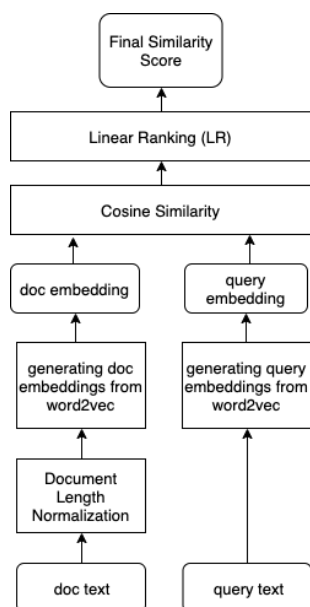


Fig. 4. The architecture of the embedding based model to rank query-document based on length normalization, word embeddings, and linear ranking.

$\cos(\vec{Q}, \vec{D})$. The reason behind considering a $\cos(\vec{Q}, \vec{D})$ score in a \bar{L} is to control it from over-ranking documents with multiple query terms but no actual relevancy. In $\text{sim}(Q|D)$ linear ranking component is completely ignored if the query count is 0, and in such a case, only the cosine score is considered to decide document rank.

Fig. 4 shows a graphical representation of the proposed model, in which the query and document similarity score is generated by a stack process of length-normalization, inferring embeddings, and ranking. A cosine score is used as a similarity measure, later combined with the linear score as shown in Equation 5. The next section presents the experiments with four different embedding spaces along with Length normalization and Linear Ranking.

IV. EXPERIMENTS

We compare the performance of normalized ranking function with a baseline based on *word2vec*, which is not utilizing any normalization techniques beforehand. We have also utilized the Dual Embedding Space Model with additional experiments [25]. We have not performed any assessment of our approach over any other term-frequency based baselines, as our primary goal is to study the impact of document length and linear ranking on top of simple embeddings based methods.

In our experiments, we considered every document in the dataset as a candidate for each query, which is aligned with the traditional approach of IR, where the model needs to retrieve a relevant set of documents from a single collection.

A. Datasets

We have used the TREC Blog06 dataset for experiments [13]. The Blog06 dataset is a collection of Blog feeds, Permalinks, and Homepage documents covering a wide area of topics such as news, sports, politics, health, and entertainment. The University of Glasgow collected this dataset for 11 weeks from December 2005 to February 2006. The combined size of collected feeds and permalinks documents is around 3 million. For assessment purposes, a set of 50 queries with a relevant judgment has also provided with the dataset.

B. word2vec Model

We trained a Skip-gram based *word2vec* model with parameter settings of 200 dimensions, 100-word count, windows size of 10, and negative sampling of 5. We used around 500,000 Blog06 feed documents for training. Before training, we performed cleaning by removing unnecessary HTML tags and stop words. We also normalized cleaned text by lower-casing all text content and replacing tokens such as “UK” with “england” and “US/USA” to “united states of america”. Moreover, we also stripped out non-English text from documents and eliminated documents with non-English content. There are two variants of the *word2vec* model, Skip-gram and CBOW. However, we train the Skip-gram model only considering the applicability of the proposed approach over the CBOW model also, as both models produce qualitatively and quantitatively similar embeddings.

C. Length Normalization and Linear Ranking

Document Length Normalization is a core component of our ranking function. While measuring the cosine similarity between document centroid and query vector, we first normalized it with Pivoted or Cosine methods. In Pivoted normalization, we run our experiments on multiple combinations of Pivot and Slope value to obtain an absolute threshold. Fig. 5 shows how MAP varying at the various Pivot-Slope combinations. Also, to provide a more subjective baseline, we run the same set of experiments using the Cosine Normalization method.

During the experiment with Pivoted Normalization, the min_x parameter was set to 0.05. For Cosine, we used 0.05 in D-OUT + Q-OUT and D-OUT + Q-IN. For D-IN + Q-IN and D-IN + Q-OUT, we have used 0.1.

Experimenting with different min_x values makes sense as each embedding space generates different sets of vectors and so its interaction gives different results. Moreover, we have also implemented various min_x combinations in the Pivoted Normalization approach, selecting 0.05 as a threshold after parameter switching. We have not assessed a min_x parameter comprehensively. However, this parameter is completely dynamic, and its value depends on the vocabulary size of the dataset. Therefore, parameter switching is required to determine the optimal threshold.

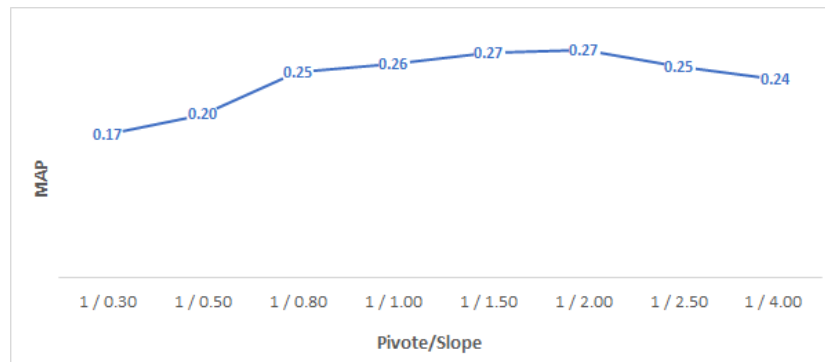


Fig. 5. MAP trend when tweaking Pivote/Slope values. Best results achieved with values between 1/1.50 to 1/2.00.

In our core ranking function, one of the parameters is the count of query words in the document, which is considered strong evidence of relevancy. We conducted experiments, including this parameter set, which improved the overall MAP considerably. Throughout all of our experiments related to Pivoted Length Normalization, including LR, we have used value 1/1.80 for Pivot and Slope.

D. IN - OUT Embeddings Space

The CBOW and Skip-gram models of *word2vec* contains two separate embedding spaces (IN and OUT). As shown in Figure 6, these embedding spaces are a set of weights from different layers. Weights for input to hidden layer are called IN embedding, and weights for hidden to output layer are called OUT embeddings [4]. By default, *word2vec* discards OUT embeddings at the end of the training. However, in this study, we keep both weights to utilize them as embedding and infer vectors from cross embedding spaces. Because, relevant few studies have found that words that appear in similar context get pushed closer to each other within IN and OUT embedding space, therefore cosine similarities in IN-IN and OUT-OUT embeddings are higher for words that are typically similar, whereas in IN-OUT cosine similarities are higher for words that often co-occur in training corpus [25].

With the motivation from the above study, we experimented with four different embedding space variants, called IN-IN, IN-OUT, OUT-OUT, and OUT-IN. For example, in the IN-IN approach, vectors for query words are taken from IN embedding, and vectors for the document are extracted from OUT embeddings. Similarly, vectors for query words are taken from IN embeddings and document words from OUT embeddings in the IN-OUT approach.

In general, IN-IN and OUT-OUT embeddings are likely to behave similarly [25]. However, in experiments, we saw further improvement in results with OUT-OUT combination outperforming any other combinations.

V. RESULTS

We performed Length Normalization based evaluation on different combinations of embedding space, as presented in

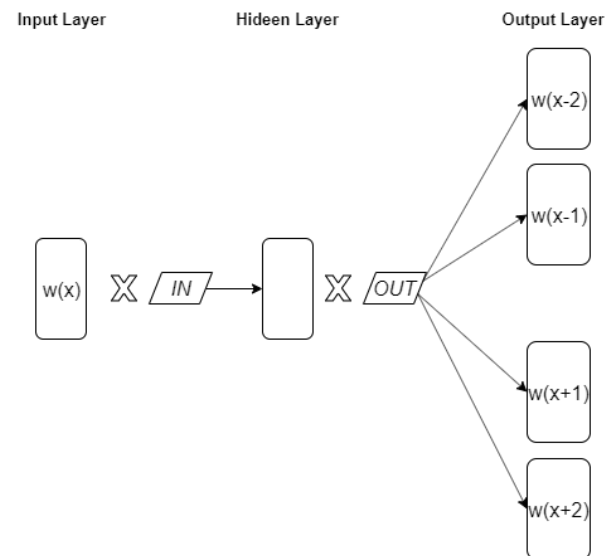


Fig. 6. The architecture of the Skip-gram model. *IN* and *OUT* are the two matrices of weight learned during training and corresponded to the IN and OUT embeddings.

Table I. We ran our experiment using a set of 48 queries out of 50 from the Blog06 dataset, ignoring two queries as an embedding for them was missing. We have significantly improved over-all baselines with Length Normalization, as shown in Table I. Combined approach of LN and LR improved performance significantly in each combination of embedding spaces. The combination of D-OUT + Q-OUT overall performed well with a MAP improving to 0.24 and 0.26 in Cosine and Pivoted Normalization. Interestingly, MAP in each combination of embedding space increases significantly with the LN function. However, the average difference between LN and LR is notably lower.

The Pivoted Normalization outperformed all other methods in terms of MAP. However, while checking its effect on the individual query, we have noticed an unstable trend with different Pivot and Slope combinations, which is shown in Fig. 7. The possible explanation to these dissimilarities is the termination of essential words from the document during

TABLE I

MAP RESULTS COMPARING LENGTH NORMALIZATION (LN) AND LINEAR RANKING (LR) WITH BASELINE. LN, ALONG WITH LR USING D-OUT + Q-OUT, PERFORMS SIGNIFICANTLY BETTER OVER ALL BASELINES. THERE IS ALSO A SIGNIFICANT IMPROVEMENT OVER BASELINE, EVEN WITHOUT THE LR COMPONENT.

Embedding Space	Baselines		LN		LN + LR	
	Simple	LR	Cosine	Pivoted	Cosine	Pivoted
D-IN + Q-IN	0.09	0.10	0.16	0.21	0.18	0.23
D-IN + Q-OUT	0.10	0.08	0.19	0.22	0.20	0.22
D-OUT + Q-OUT	0.20	0.21	0.24	0.26	0.30	0.31
D-OUT + Q-IN	0.13	0.13	0.16	0.19	0.21	0.20

Length Normalization. Tweaking parameter min_x can solve this problem to some extent.

Considering a count of query words in the document is also a significant factor. We performed a set of experiments with a linear ranking over embeddings space combined with normalization techniques (Table I). Improvement with LR is not significant in a combination of each embedding space. However, it has performed well with D-OUT and Q-OUT when combined with LN. In Table I with D-OUT & Q-OUT, we achieved a MAP of 0.31 with Pivoted Length Normalization and LR, resulting in an improvement of approx. 47% over aligned baselines.

We have not reported any experiments using term-frequency based methods such as BM25, as our primary goal is to study the possible impact of Length Normalization over embedding based methods. Instead, we used an embedding based method without normalization as our baseline.

VI. CONCLUSION

In this paper, we investigated the problem of lower-ranking of long documents in embeddings based methods. We presented a *word2vec* based embedding model with Length Normalization of documents and performed experiments with two different normalization techniques, Cosine and Pivoted, and found that Pivoted normalization improves MAP significantly. Based on our results, we also cannot ignore the fact that the presence of the query term in the document is strong evidence of relevancy, and combining it with LN can improve the ranking. To implement this idea, we have also applied LR along with LN. With these combined techniques, we achieved up to 47% improvement over baseline.

However, Length Normalization can cause a change of Average Precision for an individual query when altering Pivot and Slope values. One possible reason for this could be the loss of essential terms after normalization as it considers term-frequency and not a semantic relation.

REFERENCES

- [1] P. B. Richard Baeza-Yates and F. Chierichetti, "Essential web pages are easy to find," *Proceedings of the 24th International Conference on World Wide Web*, vol. Pages 97-107, 2015.
- [2] S. Robertson and H. Zaragoza, "The probabilistic relevance framework: BM25 and beyond," *Foundations and Trends in Information Retrieval*, vol. 3, no. 4, pp. 333-389, Apr 2009.

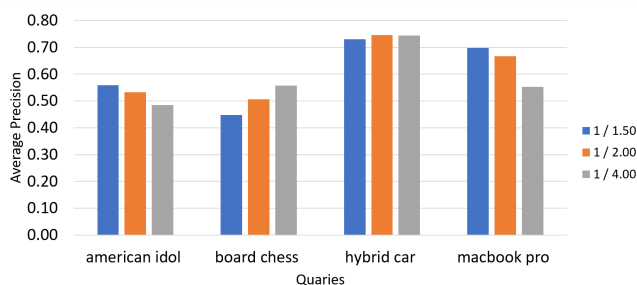


Fig. 7. Inconsistency in AP for individual queries while attempting to normalize documents by tweaking pivot & slope values.

- [3] T. Mikolov, W.-t. Yih, and G. Zweig, "Linguistic regularities in continuous space word representations," in *Proceedings of the 2013 Conference of the North American Chapter of the Association for Computational Linguistics: Human Language Technologies*, Atlanta, Georgia, Jun. 2013, pp. 746-751. [Online]. Available: <https://www.aclweb.org/anthology/N13-1090>
- [4] T. Mikolov, I. Sutskever, K. Chen, G. Corrado, and J. Dean, "Distributed representations of words and phrases and their compositionality," in *Proceedings of the 26th International Conference on Neural Information Processing Systems - Volume 2*, ser. NIPS'13, USA, 2013, pp. 3111-3119. [Online]. Available: <http://dl.acm.org/citation.cfm?id=2999792.2999959>
- [5] Y. Qi, D. Sachan, M. Felix, S. Padmanabhan, and G. Neubig, "When and why are pre-trained word embeddings useful for neural machine translation?" *Association for Computational Linguistics*, pp. 529-535, Jun. 2018. [Online]. Available: <https://www.aclweb.org/anthology/N18-2084>
- [6] Y. Shen, W. Rong, N. Jiang, B. Peng, J. Tang, and Z. Xiong, "Word embedding based correlation model for question/answer matching," *CoRR*, vol. abs/1511.04646, 2015.
- [7] Q. V. Le and T. Mikolov, "Distributed representations of sentences and documents," *CoRR*, vol. abs/1405.4053, 2014. [Online]. Available: <http://arxiv.org/abs/1405.4053>
- [8] D. Roy, "Word embedding based approaches for information retrieval," *Seventh BCS-IRSG Symposium on Future Directions in Information Access*, pp. 1-4, 2014.
- [9] H. Zamani and W. B. Croft, "Embedding-based query language models," in *Proceedings of the 2016 ACM International Conference on the Theory of Information Retrieval*, ser. ICTIR '16, New York, NY, USA, 2016, pp. 147-156. [Online]. Available: <http://doi.acm.org/10.1145/2970398.2970405>
- [10] —, "Estimating embedding vectors for queries," in *Proceedings of the 2016 ACM International Conference on the Theory of Information Retrieval*, ser. ICTIR '16, New York, NY, USA, 2016, pp. 123-132. [Online]. Available: <http://doi.acm.org/10.1145/2970398.2970403>
- [11] G. Zuccon, B. Koopman, P. Bruza, and L. Azzopardi, "Integrating and evaluating neural word embeddings in information retrieval," in *Proceedings of the 20th Australasian Document Computing Symposium*, ser. ADCS '15, New York, NY, USA, 2015, pp. 12:1-12:8. [Online].

- Available: <http://doi.acm.org/10.1145/2838931.2838936>
- [12] A. Singhal, G. Salton, M. Mitra, and C. Buckley, "Document length normalization," *Information Processing and Management*, vol. Volume 32, Issue 5, 1996.
- [13] I. Ounis, C. Macdonald, M. de Rijke, G. Mishne, and I. Soboroff, "Overview of the TREC 2006 blog track," *Proceedings of the Fifteenth Text Retrieval Conference, TREC 2006*, 01 2006.
- [14] S. Deerwester, S. T. Dumais, G. W. Furnas, T. K. Landauer, and R. Harshman, "Indexing by latent semantic analysis," *Journal of the American Society for Information Science*, vol. 41, no. 6, pp. 391–407, 1990.
- [15] D. M. Blei, A. Y. Ng, and M. I. Jordan, "Latent dirichlet allocation," *The Journal of Machine Learning Research*, vol. 3, pp. 993–1022, Mar. 2003. [Online]. Available: <http://dl.acm.org/citation.cfm?id=944919.944937>
- [16] R. Collobert and J. Weston, "A unified architecture for natural language processing: Deep neural networks with multitask learning," in *Proceedings of the 25th International Conference on Machine Learning*, ser. ICML '08, New York, NY, USA, 2008, pp. 160–167. [Online]. Available: <http://doi.acm.org/10.1145/1390156.1390177>
- [17] Y. Bengio, R. Ducharme, P. Vincent, and C. Janvin, "A neural probabilistic language model," *The Journal of Machine Learning Research*, vol. 3, pp. 1137–1155, Mar. 2003. [Online]. Available: <http://dl.acm.org/citation.cfm?id=944919.944966>
- [18] P. Bojanowski, E. Grave, A. Joulin, and T. Mikolov, "Enriching word vectors with subword information," *Transactions of the Association for Computational Linguistics*, vol. 5, pp. 135–146, 2017.
- [19] J. Pennington, R. Socher, and C. Manning, "Glove: Global vectors for word representation," *Proceedings of the 2014 Conference on Empirical Methods in Natural Language Processing (EMNLP)*, pp. 1532–1543, 2014. [Online]. Available: <http://aclweb.org/anthology/D14-1162>
- [20] J. Devlin, M. C. ands Kenton Lee, and K. Toutanova, "BERT: pre-training of deep bidirectional transformers for language understanding," *CoRR*, vol. abs/1810.04805, 2018. [Online]. Available: <http://arxiv.org/abs/1810.04805>
- [21] M. E. Peters, M. Neumann, M. Iyyer, M. Gardner, C. Clark, K. Lee, and L. Zettlemoyer, "Deep contextualized word representations," *CoRR*, vol. abs/1802.05365, 2018. [Online]. Available: <http://arxiv.org/abs/1802.05365>
- [22] S. Clinchant and F. Perronnin, "Aggregating continuous word embeddings for information retrieval," *Proceedings of the Workshop on Continuous Vector Space Models and their Compositionality*, pp. 100–109, 2013.
- [23] S. Balaneshin-kordan and A. Kotov, "Embedding-based query expansion for weighted sequential dependence retrieval model," in *Proceedings*
- [29] F. Diaz, B. Mitra, and N. Craswell, "Query expansion with locally-trained word embeddings," *CoRR*, vol. abs/1605.07891, 2016. [Online]. Available: <http://arxiv.org/abs/1605.07891>
- of the 40th International ACM SIGIR Conference on Research and Development in Information Retrieval, ser. SIGIR '17, New York, NY, USA, 2017, pp. 1213–1216. [Online]. Available: <http://doi.acm.org/10.1145/3077136.3080764>
- [24] D. Ganguly, D. Roy, M. Mitra, and G. J. Jones, "Word embedding based generalized language model for information retrieval," in *Proceedings of the 38th International ACM SIGIR Conference on Research and Development in Information Retrieval*, ser. SIGIR '15, New York, NY, USA, 2015, pp. 795–798. [Online]. Available: <http://doi.acm.org/10.1145/2766462.2767780>
- [25] B. Mitra, E. T. Nalisnick, N. Craswell, and R. Caruana, "A dual embedding space model for document ranking," *CoRR*, vol. abs/1602.01137, 2016. [Online]. Available: <http://arxiv.org/abs/1602.01137>
- [26] D. Roy, D. Ganguly, M. Mitra, and G. J. F. Jones, "Representing documents and queries as sets of word embedded vectors for information retrieval," *CoRR*, vol. abs/1606.07869, 2016. [Online]. Available: <http://arxiv.org/abs/1606.07869>
- [27] M. K. Huth2 and Christopher, "Evaluating the impact of word embeddings on similarity scoring in practical information retrieval," *Zenodo*, pp. 585–596, 2017.
- [28] M. J. Kusner, Y. Sun, N. I. Kolkin, and K. Q. Weinberger, "WMD: From word embeddings to document distances," *Proceedings of The 32nd International Conference on Machine Learning*, vol. 37, pp. 957–966, 2015.
- [30] A. Imani, A. Vakili, A. Montazer, and A. Shakery, "Deep neural networks for query expansion using word embeddings," *CoRR*, vol. abs/1811.03514, 2018. [Online]. Available: <http://arxiv.org/abs/1811.03514>
- [31] M. Grbovic, N. Djuric, V. Radosavljevic, and N. Bhamidipati, "Search retargeting using directed query embeddings," in *Proceedings of the 24th International Conference on World Wide Web*, ser. WWW '15 Companion, New York, NY, USA, 2015, pp. 37–38. [Online]. Available: <http://doi.acm.org/10.1145/2740908.2742774>
- [32] P.-S. Huang and J. Gao, "Learning deep structured semantic models for web search using clickthrough data," *ACM International Conference on Information and Knowledge Management (CIKM)*, October 2013. [Online]. Available: <https://www.microsoft.com/en-us/research/publication/learning-deep-structured-semantic-models-for-web-search-using-clickthrough-data/>
- [33] A. W. Gerard Salton and C.S.Yang, "A vector space model for information retrieval," *Journal of the American Society for Information Science*, vol. 18(11), pp. 613–620, November 1975.

Facial Recognition using Convolutional Neural Networks and Supervised Few-Shot Learning

Rafael Gallardo García, Beatriz Beltrán, Darnes Vilariño, and Rodolfo Martínez

Abstract—The paper presents a feature-based face recognition method. The method can be explained in two separated processes: A pretrained CNN-Based face detector looks for faces in images and return the locations and features of the found faces, this face detector will be used to train the models for the classifiers and then will be used to find unknown faces in new images. The used classifiers are: K-Nearest Neighbors, Gaussian Naive Bayes and Support Vector Machines. Each model will be trained with a different quantity of training examples in order to obtain the best version of the method. When the models are ready, each classifier will try to classify the faces with the previously trained models. The accuracy of each classifier in few-shot face recognition tasks will be measured in Recognition Rate and F1 Score, a comparative table of the results is presented. This paper has the goal to show the high accuracy achieved by this method in datasets with several individuals but few examples of training.

Index Terms—Convolutional neural network, facial recognition, artificial vision, few-shot learning.

Fig. 2 shows the used structure for tests before being analyzed for the facial recognition system, Fig. 3 shows the output after performing the method over the Fig. 2.

I. INTRODUCTION

ARTIFICIAL vision is one of the artificial intelligence disciplines which tries to develop, improve and research new methods through which computers are able to acquire, process, analyze and understand real-world images in order to get numerical or symbolic information that can be more easily processed by computers. The face recognition is one of the most common applications of the Biometric Artificial Intelligence, a facial recognition system is capable of identify or verify persons in digital images or videos. Face recognition technology can be used in wide range of applications such as identity authentication, access control, and surveillance. A face recognition system should be able to deal with various changes in face images [1].

II. RELATED WORK

One of the first and most successful template matching methods is the eigenface method [2], which is based on the Karhunen Loeve transform (KLT) or the principal component analysis (PCA) for the face representation and recognition.

Manuscript received on June 24, 2019, accepted for publication on August 29, 2019, published on December 30, 2019.

The authors are with the Benemérita Universidad Autónoma de Puebla, Facultad de Ciencias de la Computación, Mexico (e-mail: rafael.gallardo@alumno.buap.mx, {bbeltran,darnes,beetho}@cs.buap.mx).

Every face image in the database is represented as a vector of weights, which is the projection of the face image to the basis in the eigenface space [1]. Usually the nearest distance criterion is used for face recognition. Guodong *et al.* [1] focused on the face recognition problem and showed that the discrimination functions learned by SVMs can give much higher recognition accuracy than the popular eigenface approach [2] working on larger datasets.

Convolutional Neural Networks (CNNs) have taken the computer vision community by storm, significantly improving the state of the art in many applications. One of the most important ingredients for the success of such methods is the availability of large quantities of training data [3]. Parkhi *et al.* [3] traversed through the complexities of deep network training and face recognition to present methods and procedures to achieve comparable state of the art results on the standard LFW and YTF face benchmarks.

King Davis, developed a frontal face detector [4] using *Histogram of Oriented Gradients (HOG)* and *linear SVMs*, this HOG+SVM method achieved good results in frontal face detection but it don't detect faces at odd angles. King Davis also included a CNN-based face detector on his *Dlib library* [4] which is slower but better detecting faces in all angles, this face detector cannot perform face recognition by itself.

III. STRUCTURE OF THE FACE RECOGNITION METHOD: A THEORETICAL APPROACH

This paper describes a feature-based face recognition method, in which the features are derived from the intensity of the data without assuming any knowledge of the face structure. The feature extraction model is biologically motivated, and the locations of the features often corresponds to face's landmarks, this could be nose, mouth, eyes, eyebrows and chin. In the presented method, a CNN search and extracts face's landmarks and then a classifier will try to find the person who that face's landmarks belongs.

A. Convolutional Neural Networks

CNNs are a class of artificial neural networks where the used neurons are very similar to the biological neurons that corresponds to the receptive field of the *primary visual cortex (V1)* [5], this means that were inspired by biological processes [5], [6]. The CNN are a regularized variation of the *multilayer perceptrons*, this means that a CNN is a fully

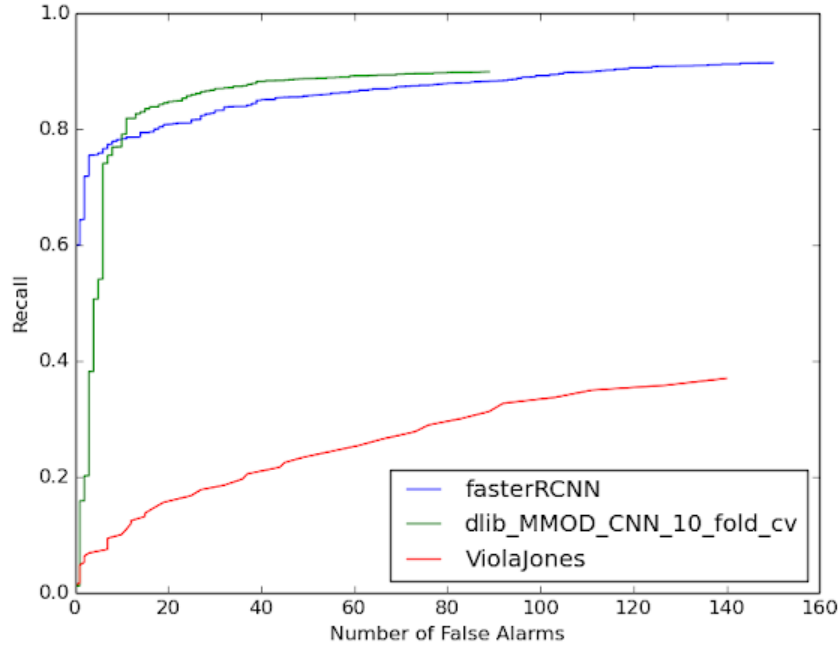


Fig. 1. CNN-MMOD on Fddb challenge.

connected network, that is, each neuron in one layer is connected to all neurons in the next layer [7], a regularized network has some magnitude measurement of weights on the loss function. Convolutional neurons works over bidimensional matrices and the extraction of the features is realized by processors that operate over the image data, the output of each convolutional neuron is:

$$y_j = g(b_j + \sum_i k_{i,j} \otimes y_i), \quad (1)$$

where the output y_i of a neuron j is a matrix that were calculated with the linear combination of the y_i neuron's output on the previous layer, each of this neurons is operated with a convolutional kernel $k_{i,j}$, this quantity is added to a b_j influence. Then, the outputs are activated with an g non-linear activation function. The convolutional operator has the function of filter the given image and transform the input data in such way that the important features become more relevant at the output [6]. Once the feature extraction is complete, the *classification neurons* will try to classify the found features with base on the imposed rules by the previous training. The behavior of this neurons is similar to the *multilayer perceptron's* neurons, and is calculated as follow:

$$y_j = g(b_j + \sum_i k_{i,j} \bullet y_i), \quad (2)$$

where the y_j output of a j neuron is calculated with the linear combination of the y_i neuron's output on the previous layer multiplied for a W_{ij} weight, the output of this operations is

added to the influence factor b_j , then is activated with a g activation function.

B. K-Nearest Neighbors Classification

This is a non-parametric supervised learning method where the estimations are based on a training dataset and all computation is deferred until classification. k-NN estimate the density function $F(x|C_j)$, where x are the predictors and C_j is each class. The training examples are vectors in a multidimensional characteristic root, each example is described in terms of p attributes considering q classes for classification. A partitioning of the space is performed in order to separate regions and it labels. A point e belongs to C if this class is the most frequent in the k closest training examples, this is:

$$d(x_i, x_j) = \sqrt{\sum_{r=1}^p (x_{ri} - x_{rj})^2}, \quad (3)$$

the training phase stores the eigenvectors and the labels of the classes of the training examples, this is: for each $\langle x, f(x) \rangle$, where $x \in X$, this example should be added to the examples structure. In the classification phase, on an example x_q that will be classified, being x_1, \dots, x_n the k nearest neighbors to x_q in the training examples, then the output will be:

$$\hat{f} \leftarrow \operatorname{argmax}_{v \in V} \sum_{i=1}^k \varsigma(v, f(x)), \quad (4)$$

TABLE I
SCORES OF THE FACIAL RECOGNITION METHOD.

Training Shots	Algorithm	RecognitionRate	Precision	Recall	F1
1-Shot	k-NN	99.15	0.991	0.991	0.991
	SVM	N/A	N/A	N/A	N/A
	GNB	99.09	0.921	1	0.959
2-Shots	k-NN	99.159	0.991	0.991	0.991
	SVM	100	0.915	1	0.955
	GNB	99.159	0.921	1	0.959
3-Shots	k-NN	99.15	0.991	0.991	0.991
	SVM	100	0.915	1	0.955
	GNB	99.159	0.921	1	0.959
4-Shots	k-NN	100	0.991	1	0.995
	SVM	100	0.915	1	0.955
	GNB	100	0.922	1	0.959
5-Shots	k-NN	100	0.991	1	0.995
	SVM	100	0.915	1	0.955
	GNB	100	0.922	1	0.959
6-Shots	k-NN	100	0.991	1	0.995
	SVM	100	0.915	1	0.955
	GNB	100	0.922	1	0.959
7-Shots	k-NN	100	0.991	1	0.995
	SVM	100	0.915	1	0.955
	GNB	100	0.922	1	0.959
8-Shots	k-NN	100	0.991	1	0.995
	SVM	100	0.922	1	
	GNB	100	0.922	1	0.959
9-Shots	k-NN	100	0.991	1	0.995
	SVM	100	0.922	1	0.959
	GNB	100	0.922	1	0.959
10-Shots	k-NN	100	0.991	1	0.995
	SVM	100	0.929	1	0.963
	GNB	100	0.937	1	0.967

where $\varsigma(a, b) = 1$ if $a = b$ and 0 in other cases. If $k = 1$ the closest neighbor to x_i determines its value.

C. Support Vector Machines Classification

Support Vector Machines (SVM) are supervised learning models, or learning algorithms which build *non-probabilistic binary linear classifiers* that assign new examples to a category. Given training vectors $x_i \in \mathbb{R}^p$, $i = 1, n$, in two classes, and a vector $y \in \{1, -1\}^n$, the algorithm solves the following problem:

$$\min_{w, b, \zeta} \frac{1}{2} w^T w + C \sum_{i=1}^n \zeta_i, \quad (5)$$

subject to:

$$y_i(w^T \phi(x_i) + b) \geq 1 - \zeta_i, \zeta_i \geq 0, i = 1, \dots, n, \quad (6)$$

and its dual is:

$$\min_{\alpha} \frac{1}{2} \alpha^T Q \alpha - e^T \alpha, \quad (7)$$

subject to:

$$y^T \alpha = 0 \text{ and } 0 \leq \alpha_i \leq C, i = 1, \dots, n, \quad (8)$$

where e is the vector of all ones, $C > 0$ is the upper bound, Q is an n by n positive semidefinite matrix, $Q_{ij} \equiv y_i y_j K(x_i, x_j)$, where $K(x_i, x_j) = \phi(x_i)^T \phi(x_j)$ is the kernel. The training vectors are implicitly mapped into a higher dimensional space by the function ϕ . The decision function is:

$$\sum_{i=1}^n y_i \alpha_i K(x_i, x) + \rho. \quad (9)$$

D. Gaussian Naive Bayes Classification

Naive Bayes methods are supervised learning algorithms based on the application of the Bayes' theorem and by assuming the conditional independence between every pair of features given the values of the class variable, this is a naive assumption. Naive Bayes classifiers use the following classification rule:

$$P(y | x_1, \dots, x_n) \propto P(y) \prod_{i=1}^n P(x_i | y), \quad (10)$$



Fig. 2. Example of testing subset.

the previous equation could be write as:

$$\hat{y} = \arg \max_y P(y) \prod_{i=1}^n P(x_i | y). \quad (11)$$

In *Gaussian Naive Bayes*, the likelihood of the features is assumed to be Gaussian:

$$P(x_i | y) = \frac{1}{\sqrt{2\pi\sigma_y^2}} \exp\left(-\frac{(x_i - \mu_y)^2}{2\sigma_y^2}\right), \quad (12)$$

σ_y and μ_y are estimated using maximum likelihood.

IV. STRUCTURE OF THE FACE RECOGNITION METHOD: A TECHNICAL APPROACH

In general, the facial recognition method consists in a face detector, and a clustering algorithm, for experimental purposes, three different classifiers were tested, technically the *Table 1* presents a comparison between the methods, which have the

same CNN-based face detector, but its accuracy will depend on the effectiveness of each classifier when clustering and recognizing faces. Details are specified below.

A. CNN-based Face Detector

This method used a *Maximun-Margin Object Detector(MMOD)* [8] with CNN-based features. The face detector were trained with various datasets like ImageNet, PASCAL VOC, VGG, WIDER and Face Scrub, the training dataset for the face detector contains 7220 images even so, a good CNN-based face detector can be obtained with a training dataset with just 4 examples. The CNN version of MMOD tested with the 10-fold cross-validation version of the FDDB challenge [9], gives the results shown in Fig. 1. The X axis is the number of false alarms produced over a dataset with 2885 images. The Y axis is the fraction of faces found by the detector (recall). The green curve is the CNN-based MMOD trained with 4600 faces, the red curve is the old Viola Jones



Fig. 3. Example of results over the testing subset.

detector, and the blue curve is the Faster R-CNN [10] trained with 159,424 faces. The results presented on the chapter 5 were realized by running the CNN face detector over the *NVIDIA CUDA Deep Neural Network (cuDNN)* library, with a NVIDIA 960M GPU with 4GB of VRAM with Maxwell architecture and 5.x compute capability.

B. Structure of the Experiments

In order to test the facial recognition method and the accuracy of the classifiers as few-shot learning, 10 models were trained per each classifier, each model has a different quantity of training examples per subject, from 1 training example to 10 training examples per subject per classifier, 30 models at the end of training. The accuracy of the method is the result of the combination of the achieved accuracy for the CNN face detector when searching for faces in images and extracting it face's landmarks, and the achieved accuracy of the clustering algorithms when finding the class of each unknown

face. The experiments were performed over the *MIT-CBCL Face Recognition Database* [11], this database contains face images of 10 subjects in high resolution, including frontal, half-profile and profile views. Test images have a size of 115x115 pixels.

V. EXPERIMENTAL RESULTS

Table I presents the obtained scores with the three different classifiers, when using the CNN-MMOD as face detector and the classifiers as face recognizer, this experiments were performed over a subset of the *MIT-CBCL Face Recognition Database* [11], the results on the *table 1* belongs to an experiment performed over a subset of 119 known faces with 11 unknown faces to be recognized.

VI. CONCLUSIONS

The experiments are clear, the presented facial recognition method can be considered as a success, the three used

classifiers gave a $> 99\%$ of Recognition Rate and a F1 score > 0.9 , mixing the accuracy of a CNN-based face detector with face clustering techniques gives very interesting results.

Focusing on the difference between the classifiers, it's easy to see that the k-Nearest Neighbors classifier has a recognition rate similar to the other classifiers but it's noticeably better in Recall and Precision scores, in consequence it is better in the F1 score. Surprisingly, the quantity of training examples per subject did not greatly affect the accuracy of the facial recognition method. k-Nearest Neighbors was notably better while recognizing known people and was better while labeling faces as "unknown", Support Vector Machines and Gaussian Naive Bayes versions of the facial recognition method gave excellent results while recognizing known faces but gave disappointing results while trying to identify unknown faces, labeling faces as "known" and giving false positives.

Execution times of the CNN-MMOD face detector is very slow when it runs on a Intel Core i7-6700HQ CPU, with a 4.5 FPS average, but this FPS improve if the face detector runs over a CUDA GPU, reaching up to 200 FPS, this FPS makes CNN-MMOD+k-NN a feasible method for real-time facial recognition. k-NN reaches high accuracy scores, both in F1 and in Recognition Rate, with feasible times by using parallel programming.

Future work include optimizing the execution time to obtain a faster facial recognition system, this time optimization consists in the optimization of the classifiers and the CNN. Also, this facial recognitions method will be tested in different type of cameras and places in order to evaluate it performance on the real world.

VII. ACKNOWLEDGMENTS

Credit is hereby given to the Massachusetts Institute of Technology and to the Center for Biological and

- [5] D. H. Huvel and T. N. R. f. a. Wiesel, "and functional architecture of monkey striate cortex," *The Journal of Physiology*, vol. 195, pp. 215–243, 1968.

Computational Learning for providing the database of facial images.

Also thanks to the vice-rectory of research and postgraduate studies (VIEP) of the Benemérita Universidad Autónoma de Puebla for the project: *Metodología basada en gramáticas para la detección de eventos anómalos por medio de redes complejas en tiempo real*.

REFERENCES

- [1] G. Guodong, Z. L. Stan, and C. Kapluk, "Face recognition by support vector machines," in *Proceedings of the Fourth IEEE International Conference on Automatic Face and Gesture Recognition*, 1970.
- [2] M. A. Turk and A. P. Pentland, "Eigenfaces for recognition," *Cognitive Neuroscience*, vol. 3, pp. 71–86, 1991.
- [3] O. M. Parkhi, A. Vedaldi, and A. Zisserman, "Deep face recognition," in *Proceedings of the British Machine Vision Conference (BMVC)*, X. Xie, M. W. Jones, and G. K. L. Tam, Eds. BMVA Press, September 2015, pp. 41.1–41.12. [Online]. Available: <https://dx.doi.org/10.5244/C.29.41>
- [4] E. D. King, "Dlib-ml: A machine learning toolkit," *Journal of Machine Learning Research*, 2009.
- [6] M. Matusugu, M. Katsuhiko, and M. Yusuke, "Subject independent facial expression recognition with robust face detection using a convolutional neural network," *Neural Networks*, vol. 16, pp. 555–559, 2003.
- [7] L. Hongxin, S. Xiaorong, and R. F.-N. Haibing, "Joint convolutional neural networks for face detection and attribute recognition," in *9th International Symposium on Computational Intelligence and Design*, 2016.
- [8] E. D. King, "Max-margin object detection," arXiv [cs.CV] 1502.00046, 2015.
- [9] V. Jain and E. Learned-Miller, "FDDB: A benchmark for face detection in unconstrained settings," Dept. of Computer Science, University of Massachusetts, Amherst, Technical Report UM-CS-2010-009, 2010.
- [10] H. Jiang and E. Learned-Miller, "Face detection with the faster R-CNN," arXiv [cs.CV] 1606.03473, 2016.
- [11] B. Weyrauch, J. Huang, B. Heisele, and V. Blanz, "Component-based face recognition with 3D morphable models," in *First IEEE Workshop on Face Processing in Video*, 2004.

Zero-Shot Learning for Topic Detection in News Articles

Björn Buchhold and Jörg Dallmeyer

Abstract—We present a method to detect topics in news articles. The topics of interest are each represented by a descriptive document. We train a model that can be seen as a similarity function between such a descriptive document and a news article. Our model is a neural network that operates on two kinds of inputs. (1) The full texts of the descriptive documents and the news articles are passed through the same recurrent encoder network and then the distance of the resulting encodings is taken. (2) Our proprietary NLP pipeline and knowledge base are used to recognize named entities and significant keywords and we compute features based on their overlap for a descriptive document and a news article. Our model finally combines the encoding distance with the overlap features and acts as a binary classifier. We evaluate and compare several model configurations on two datasets, a large one automatically created from Wikipedia and a smaller one created manually.

Index Terms—Topic detection, zero-shot learning, NLP, deep learning.

I. INTRODUCTION

THIS WORK focuses on a special kind of topic detection. Think of a topic that emerges in news, for example the *Diesel emissions scandal*. It can be of great value to recognize all news articles that talk about such a topic. In combination with the identification of named entities, users can then track if a company they are interested in (maybe because they own its stock; maybe because the company is one of their suppliers, customers or a competitor) is mentioned in the context of the scandal. Likewise, the other way round is useful, i.e., to see all topics that are emerging for a selection of interesting entities.

This task is challenging in several ways: topics are often not mentioned literally in the news but inferred by the content and the involved entities. In the example of the *Diesel emissions scandal* not all three words have to be mentioned in this order for a news article to be talking about the topic, however, simply looking for the individual words, variants and synonyms is hopelessly imprecise. Furthermore, classic approaches to topic modeling are not applicable here either, because they all build models around a corpus that already contains the topics of interest. However, it is the very essence of news, that entirely new topics emerge and are reported on.

We want to improve an existing system performs for news monitoring.

Manuscript received on June 18, 2019, accepted for publication on September 4, 2019, published on December 30, 2019.

The authors are with CID GmbH, Germany (e-mail: {b.buchhold, j.dallmeyer}@cid.com; web: <http://cid.com>).

It performs web crawling, content extraction, and several NLP tasks including the identification of named entities from customizable knowledge graphs. Enriched documents are stored in an indexing system with support for near real-time analyses on millions of documents. So far, the system supports two approaches for modeling and detecting topics, which tackle the problem from different perspectives.

So-called Hot Topics (HT) are computed on a document analysis set (e.g., over the last three days). Entities and keywords that are significantly more frequent in the analysis set than in a reference set are grouped according to co-occurrence. Such a group then represents an HT. An HT is dynamic, unsupervised and has no meta information and no proper name.

The second approach consists of so-called Supervised Topics (ST). STs are manually defined topics consisting of a set of weighted keywords and named entities. With STs it is possible to detect topics in future news (e.g., earthquakes or C-Suite changes). STs use a linear retrieval model of weighted aspects. However, the fine-tuning of selected aspects, their weights and acceptance thresholds requires domain knowledge.

The contribution of this paper is a novel approach for the kind of topic detection described in the beginning. We base it on descriptive documents for topics that can then be associated to news articles using a combination of traditional NLP and Deep Learning: We view our problem as instance of zero-shot learning. We train an encoder that infers abstract representations from text documents and our NLP pipeline extracts named entities and keywords from them. For a descriptive topic document and a news article these abstract representations and the overlap in extracted NLP features are used to calculate their similarity. New topics can be introduced to the system without re-training the underlying model. We show how training data for this problem can be gathered automatically, introduce suitable models and evaluate them in different experiments.

We make use of the work by the Wikipedia¹ community, whose manual curation yields up-to-date topic documents with very high quality. These documents have a unique identifier, a description and representations in several languages. This gives control for selection and customization of relevant topics for users with diverse interests.

¹https://en.wikipedia.org/wiki/Portal:Current_events

Section II reviews related work. In Section III the fundamental task of generating training data for the learning step is discussed. We give a model description in Section IV. Section V shows the experimental setup and results, as well as variants of the model. We conclude in Section VI and give ideas for future work.

II. RELATED WORK

This work studies a new problem as motivated in Section I. Thus, we cannot compare our results to those for well-studied problems with prominent datasets. Nevertheless, there are several fields of work that are closely related. We distinguish three categories: (1) Literal approaches and the well-established task of Named Entity Identification where topics could be treated as just another kind of entity; (2) Topic Models, which are well-studied but not compatible with the way we want to define topics of interest; (3) Classification via zero-shot learning in other domains, i.e., machine learning approaches that assign classes, even if there never was any training data for a particular class, e.g., systems for face recognition.

A. Literal Topic Identification

We could treat our problem as just another kind of Named Entity Identification with topics as entities to identify. In our experiments, this approach did not yield good results. While this works out in some cases, the limitations quickly become apparent.

Literally matching *Brexit* in news articles and maybe adding other strong signal words, e.g., *Brexit* works pretty well. A counter example is the *China–United States trade war* and a news article about the implementation of certain tariffs. It is conceivable that there is no literal mention of the trade war, even though the article is relevant. When too liberal signal words are added, the precision drops significantly.

B. Topic Models

Topic Models are well-researched. In a sense, they induce a soft clustering (where a document can belong to 30% to topic A and to 70% to topic B) on a document collection. Thus, these topics are usually purely statistical and abstract, i.e., they do not carry a name nor necessarily correspond to an intuitive topic.

The most prominent approaches to topic modeling include Latent Dirichlet Allocation (LDA) [1] and Probabilistic Latent Semantic Analysis (pLSA) [2]. Usually, these approaches are given a number of topics to discover. They then assume that documents are produced by the following generative process: For each word, first one topic of the current document is chosen, and then a word is picked from all possible words with the conditional probability given the chosen topic. Given this generative model, the topics are assigned to a document according to a maximum likelihood estimate.

An important difference to our problem is that only the number of topics to infer is given and a document collection is fit accordingly. The resulting topics are not necessarily interpretable in an intuitive way. More importantly though, newly emerging topics do not work at all unless the whole model is fit to a document collection in which the topic plays a significant role.

Labeled LDA (LLDA) [3] is an extension of LDA where not only the number of topics to infer is given but also concrete topic labels. This overcomes the problem of nameless topics that do match expected kinds of topics. However, this extension is also not applicable to our scenario, because it still has to be fit on a corpus that already contains all topics of interest.

In [4] the static Dewey Decimal Classification (DDC) is applied for thematic classification of short text snippets, e.g., tweets. The work deals with the need for fast and accurate classification in scenarios of lexical sparseness. The best classification results were achieved by a combination of a Support Vector Machine (SVM) and a Neural Network (NN). The SVM used *tf-idf* term weights, n-grams and LDA topic features. The NN uses fastText [5] in combination with nodes with an activation function signaling membership to LDA based topics.

The training is done on DDC classified German documents. The derived model is used to explore the distribution of DDC topics in a pool of text documents. In contrast to that, we focus on changes in the collection of topics.

The Topic Detection and Tracking (TDT) study [6] features several related tasks. However, most of them are similar to work discussed previously: Entire text corpora are segmented into topical clusters, or well-known topics are tracked throughout news. One task for *On-Line New Event Detection and Tracking* [7] is more closely related to our work, the major difference being the absence of our descriptive documents. However, the approaches to online detection still treat the problem as a clustering problem, where unassigned documents form their own topical clusters. The study concludes that “Online detection cannot yet be performed reliably”. In a sense, our problem is a slightly simplified variant of this hard task.

C. Zero-Shot Learning

The problem looks like yet another instance of text classification. However, since new topics can occur at any time, the problem becomes a lot more intricate. We cannot expect to re-train the model every time a new topic occurs, but even more importantly, we have no training data to do so. Thus, the problem can be seen as instance of zero-shot learning, i.e., the model has to predict topics for which it has not once seen explicit examples to learn from. This is sometimes also known as zero-data learning [8].

In [9], the authors present a framework for zero-shot learning. Just like in our scenario, not all target classes

occur in training data. However, there is a difference. Their classification targets are from a semantic knowledge base, which describes animals by features such as *is it furry?* or *does it have a tail?*. The authors then design their classifier to predict a feature vector with scores for the features from the semantic knowledge base. The class with the closest feature vector is then predicted as the result. In their case study, they have trained models to produce such vectors from fMRI images, i.e., from people's neural activity.

In our case, we do not have such a semantic knowledge base with all, and especially upcoming, topics. Our problem is more closely related to face recognition and face verification, where new faces are added without manually deriving feature representations (and also without re-training the models). A popular system for face recognition is FaceNet [10], which has inspired much of our work. The authors train a neural-network encoder that produces a high-dimensional encoding vector for each known image and for the input image. A threshold for distances in euclidean space between these encodings can then be used to accept or reject matches. Our work follows the same approach where we replace the image encoder with a textual encoder. We augment this model by features derived from our NLP pipeline. However, it is possible to train a textual encoder, so that the L2-distance between encodings already identifies topics fairly well. In our experiments in Section V we quantify this and compare the approach to other variants and to our full model.

An important part of FaceNet is that the encoders are trained using a specialized triplet-based loss function that operates on triples of anchor image, positive example and negative example.

The loss function then requires the encoding of the anchor to be closer to the encoding of the positive example than to the encoding of the negative example by a given margin. We can also use this loss function to train our textual encoders, however with slightly worse results than by simply training them in a binary classifier.

III. ACQUISITION OF TRAINING DATA

For most practical applications of machine learning, finding a suitable model is just one part of the problem. In absence of an established dataset, the acquisition of training data is often the biggest challenge to overcome. For our use case, we are not aware of any suitable dataset and manual creation would be extremely tedious and costly. Hence, we have designed a system to automatically retrieve such data from publicly available information, in particular from Wikipedia.

Our source are Wikipedia articles about the current events for a given date. These pages exist for every day since 1994, and since 2003 they adhere to a format that is very useful for our purpose. For important events, editors quickly generate dedicated Wikipedia articles and link to them whenever there are new developments. The event essentially becomes a topic that occurs in the news for a few days, weeks, or possibly many years (e.g., major political conflicts).

In Figure 1 we illustrate how we extract topics for a specific day: We ignore category headlines and work with the bullet points. If (and only if) a top-level bullet point does have subordinate bullet points, we take the linked Wikipedia articles of the top-level bullet as topic. The contents of these linked Wikipedia articles can then be used as descriptive documents. Further, every external link under the bullet point refers to a relevant news article. We extract the body text behind those links and regard the associated topic as a positive training example.

These connections to external news articles turn out to be very reliable training data. However, many of them have to be skipped: The links may no longer be working, they may be unavailable for automatic retrieval, the news articles may be in some other language, etc. To gather additional positive topic-news pairs, we also visit the Wikipedia articles about the topics themselves and extract the References section (which contains links to documents outside Wikipedia). This gives us numerous positive pairs, albeit with some uncertainty (a reference may be evidence for a very specific statement in the article and is not necessarily relevant to the entire topic). Experiments have shown that their inclusion does significantly more good than harm.

We treat our problem as pairwise classification problem and thus need negative training examples, i.e., news articles and topics which should not be detected for them. One way is to sample these pairs at random. For each positive pair of topic and news article, we can produce negative pairs with the same topic and a random other article and negative pairs with the same article and a random topic. We choose to make the number of negative pairs configurable for two reasons: (1) Taking all negative pairs for thousands of topics and tens to hundreds of thousands of articles results in way too many negative pairs to handle efficiently. (2) Such negative pairs are not perfectly reliable. For example, a news article may be linked to the topic *Dismissal of James Comey* but it may also be relevant to topics like *Presidency of Donald Trump*. For these two reasons we limit (and thus essentially under-sample) the number of pairs of negatives. We end up with sufficient data and while we may still pick a false negative pair at random, the chances are very low. At the very least we will have significantly more correct positive pairs (and correct negative pairs) than problematic pairs.

While the above strategy already produces a functioning model, a problem remains. Negative pairs picked at random have a tendency to be too easy to distinguish from positives ones, because they may be from very different domains. The resulting models do not work well in practice, despite achieving very high accuracy (and also great performance w.r.t. other metrics like F-measure) on our test data. As an example, assume a news article about the topic *2019 elections in India*. It is easy to accept that topic and reject completely unrelated ones like *Gun laws in New Zealand*. However, it is much harder to handle topics like *2014 Indian general election* or *2019 Sri Lankan presidential election* in that case, because

Business and economy

- **Nicotine marketing**
 - Michigan becomes the first state in the United States to ban the sale of flavored electronic cigarettes, which the state government says are marketed towards children. (MLive.com)

Law and crime

- **2019 Samoa assassination plot**
 - After an assassination plot to kill Samoan Prime Minister Tuilaepa A. S. Malielegaoi is foiled, police are working to extradite a Samoan man who lives in Brisbane, Australia. (Radio New Zealand)
- **2019 Hong Kong anti-extradition bill protests**
 - Hong Kong Chief Executive Carrie Lam announces the formal withdrawal of the controversial extradition bill. (Reuters)
- **Privacy concerns regarding Google**
 - Google agrees to pay a record US\$170 million penalty to settle accusations that YouTube broke the law when it knowingly tracked and sold advertisements to children, the Federal Trade Commission says. (CNN)

Politics and elections

- **Brexit**
 - The House of Commons of the United Kingdom approves a bill to block a no-deal Brexit next month, by a vote of 327 to 299. The bill instructs Prime Minister Boris Johnson to request another Brexit extension if he cannot secure a deal with the European Union in the coming weeks. (The New York Times)
 - The House of Commons rejects Prime Minister Boris Johnson's motion to hold a general election in October amid continuing political deadlock over Brexit. (BBC)

Fig. 1. An excerpt from the Wikipedia article on current events of September 9, 2019. The red boxes mark the topics we are able to extract from this. The green boxes mark corresponding relevant news articles.

these topics are semantically more closely related, yet still a wrong choice for the article. If we do not include these hard pairs, it becomes much easier to correctly classify the items of our evaluation set than to use the model in practice.

Therefore, we created a harder dataset by selecting random news articles and used early versions of our models to predict the 10 most likely topics for each. In essence, this gives us topics that are similar to the news article w.r.t. different notions of similarity but may or may not be correct. We then asked 17 judges to label the topic-article pairs. We kept labels on which enough judges agreed. In our experiments in Section V we elaborate further on this process and on our results.

IV. MODEL

We want to be able to recognize multiple topics for a single news article, hence our problem can be seen as a multi-label classification. Consequently, we design our model to take a pair consisting of a descriptive document for a topic and of a news article. Then the model acts as a binary classifier that decides to accept or reject the pair.

Our model is an ensemble Neural Network that consists of a binary classifier working with features based on entity- and keyword-overlap on the one hand, and a siamese text

encoder NN (bidirectional LSTMs with self attention) on the other hand. We illustrate this in Figure 2. The encoder NN (shown as the top part) produces high-dimensional embedding vectors that represent a descriptive topic document and a news article. Similarity or distance between these embeddings can then be used directly for classification or included in a larger model as depicted in the figure. The bottom part shows that we engineered features based on the overlap of extracted entities and keywords. The combination of manually engineered features and encoding distances is then run through a simple multilayer perceptron to produce the final classification decision. In Section V we examine how well each part performs on its own and how much we gain by putting both parts together.

A. Features Based on Entity and Keyword Overlap

The bottom part of what is shown in Figure 2 requires named entities and keywords to be extracted from text documents.

Our approach to topic detection would, in principle, work with any NLP pipeline that allows making such extractions. Nevertheless, the quality of NLP affects the overall

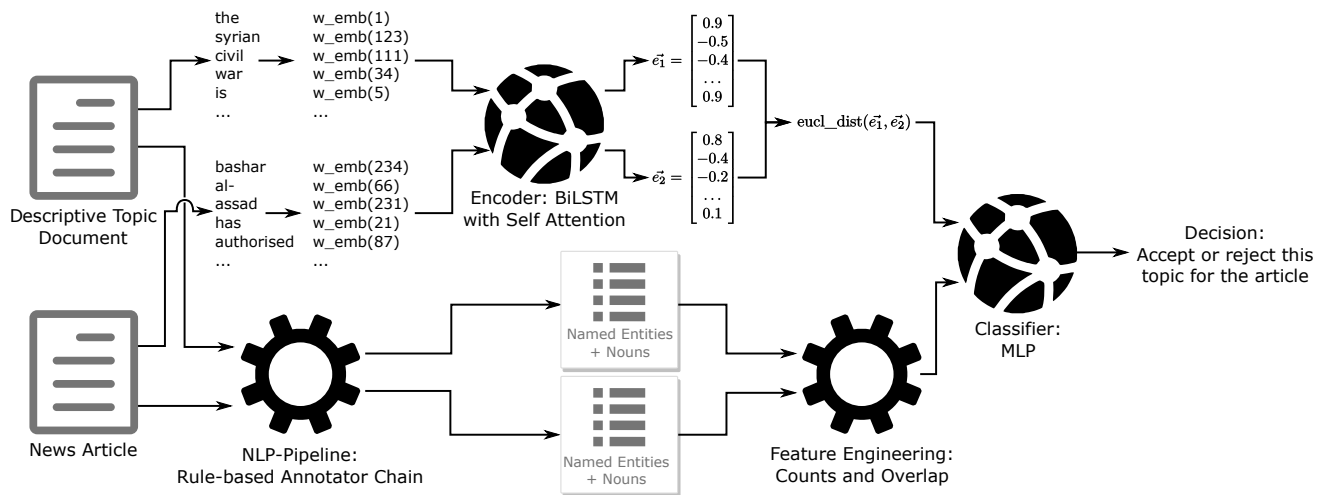


Fig. 2. A schematic view of our model. The final classification is made based on features that comprise the distance of document embeddings (top part) and manually engineered features over extracted named entities and nouns (bottom part). The LSTM and the MLP can be trained either jointly or individually.

classification result and thus we use our proprietary NLP pipeline and describe it briefly.

At first, the language is detected and the processing pipeline adapts itself to the detected language. The text is tokenized, compounds are split and a dehyphenization is done. Sentence boundaries are detected and POS tags are assigned. Lemmatization is important for extraction of proper keywords. Multi Word Expressions are detected. Named Entity Recognition (NER) is done for persons, companies, organizations and locations.

A Semantic Knowledge Graph (SKG) is used for Named Entity Identification (NEI). We build the SKG regularly using public sources (e.g., multiple Wikipedias, Wikidata, OpenStreetMap) in combination with individual sources for our clients in order to produce optimized SKGs for different use cases. Entities within the SKG have several labels, structured relationships to other entities, contextual vectors, and label-specific contexts or scores. Each piece of information found within a text document is compared to facts from the SKG in order to disambiguate multiple entities sharing the same label.

The results of our NLP pipeline are then used to extract 13 real-valued features for a pair of topic and news article: 6 features each to characterize the overlap of extracted keywords and entities and 1 feature for the cosine similarity between the average GloVe [11] embeddings over extracted keywords. For a descriptive topic document $D1$ and a news article $D2$, the six features to characterize overlap are:

- 1) the number of distinct items in $D1$
- 2) the number of distinct items in $D2$
- 3) the number of distinct items that occur in both
- 4) the sum over the $tf-idf$ values for items in $D1$
- 5) the sum over the $tf-idf$ values for items in $D2$
- 6) the sum over the $tf-idf$ values for items that occur in both

The intention behind the *idf*-normalization is that some entities and keywords are very frequent. Just because two news documents frequently mention the *USA*, that does not mean that they are about the same thing. However, if two documents mention a rather specific entity, e.g., a fugitive terrorist, this is a much stronger signal. We compute *idf* values for identified entities and keywords over a set of 5.5 million English news articles from the year 2018 that have been processed with our NLP pipeline.

In our experiments (Section V), we show that these features together, and to a lesser extent also the three groups (average embedding, keyword overlap, entity overlap) in isolation, can be used for topic detection.

B. Text Encoder

The features presented in Section IV-A are relatively rough. With this shallow form of text understanding, the models are bound to hit a quality ceiling eventually. State-of-the-art models for text understanding (e.g., for classification or translation tasks) come with enough complexity to reach much higher ceilings. The question is how to apply them to our problem, especially due to its zero-shot nature.

We follow an approach that is inspired by FaceNet [10]. Just like FaceNet uses typical models for image processing to encode anchor and input images, we use models for NLP to encode topic descriptions and the input news articles. In FaceNet and in our work, the resulting encodings can then be compared and a pair with high-enough similarity is accepted. The top part of Figure 2 illustrates this principle for our use-case.

Descriptive documents and news articles are interpreted as sequences of word embeddings, i.e., every word of the input text is replaced by a high-dimensional vector that represents its meaning. We have experimented with (1) publicly available pre-trained word embeddings, like GloVe [11] and

word2vec [12], (2) self-trained GloVe embeddings on our own corpus of news articles and (3) randomly initialized embeddings that are learned during the training of the text encoder. Differences were rather small, and in order to be flexible w.r.t. changes to lemmatization or the NLP pipeline (see Section IV-A), our experiments use randomly initialized embeddings that are learned on the fly. This also allows the model to learn an embedding for *unknown*, out-of-vocabulary words.

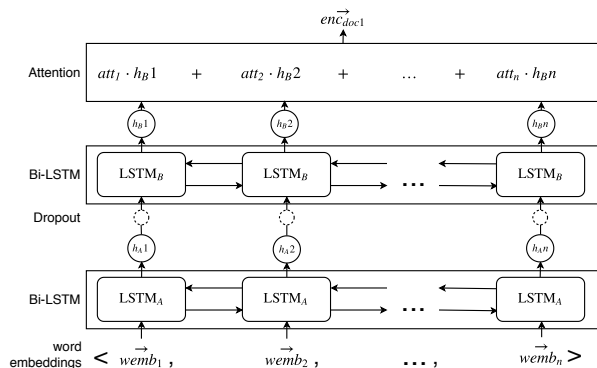


Fig. 3. Our network architecture to produce document encodings. The word embeddings pertaining to the input text are passed through two bidirectional LSTM layers. Finally, an attention layer is used to obtain the final document encoding. This layer is an implementation of the attention mechanism described in [13].

Next, the sequences of word embeddings are passed through the Neural Network depicted in Figure 3 to produce document embeddings. Finally, a pair of document encodings is compared using a simple distance function, e.g., based on euclidean distance or the cosine similarity between the two embedding vectors.

If we use the model in isolation (i.e., not as part of an ensemble), we add a fully-connected layer with one unit and sigmoid activation to find the optimal threshold to accept or reject pairs based on their distance.

We have trained these models in two ways: In the standard way, i.e., as a binary classifier that takes a pair of topic and document together with a binary true/false label, and secondly with the triplet loss that is used to train FaceNet. In our experiments using the triplet loss was not beneficial (not very harmful either) and thus we train our encoders like a standard binary classifier. This way we can use the same setup for the classifier based on overlap features, for the encoders, and for their combination. One explanation for the lack of advantages from training with the triplet loss could be that two news articles may touch the same topic but in addition to that, touch different further topics as well. News articles for the same topic can be fundamentally different from each other and are not simply variations of the same thing. In contrast, positive example images for the same face to be recognized may also be very different images, but the person depicted is the same and a perfect model might extract the same features identifying the person.

While training the model is computationally expensive, performance is not really an issue during inference. We have to compute many similarities (in particular for each news article as many as we have available topics), but the expensive computation of the encoding only has to be done once for each topic and for each news article. Hence, our topic detection scales to several thousand of topics with negligible processing times per document in comparison to the time spent in the NLP pipeline.

C. Combination

We use all features from Section IV-A and the distance of encodings from Section IV-B as input for a basic binary classifier.

We have tried several model architectures and settled for a small multilayer perceptron (MLP) with the following layers: The input features are concatenated and passed through a fully-connected layer with 10 units and ReLU activation. Then we apply batch normalization and pass the result through a fully-connected layer with 5 units and ReLU activation. Finally, we apply batch normalization again, and add a layer with one unit and sigmoid activation for prediction of the value.

We train networks that include the textual encoders, i.e., we build a joint model. We experimented with updating the encoder's weights during training and with pre-training the encoders separately and fixing their weights for training the final classifier.

V. EXPERIMENTS

In the absence of an established benchmark for our problem, it is hard to provide metrics where absolute values convey much insight. Especially the available topics and their descriptive documents have a huge impact: One aspect is granularity, our Wikipedia-based heuristic extracts topics for various battles, sieges and offensives in wars (e.g., *Manbij offensive* or *Battle of Aleppo (2012–2016)*) and for rounds in sports competitions (e.g., *2018–19 UEFA Champions League knockout phase*). Assigning the correct topics out of closely related topics is much harder than distinguishing between broader topics like *Syrian Civil War* and *Soccer*.

Another aspect is the quality of descriptive documents: Some Wikipedia articles may be perfectly suited for our approach, others only contain tabular data or compile links to other articles.

A. Data

For our experiments we used data from two sources: (1) automatically retrieved positive examples for news articles about topics as described in Section III which we augment by random topic-news pairs as negative examples and (2) manually labeled topic-news pairs that were pre-selected to be *difficult*.

1) *Easy & Large*: The process described in Section III, with picking 10 negative pairs per positive example, gives us 78,745 positive and 787,450 negative pairs about 4,390 distinct topics and 79,042 distinct news articles. We shuffled the data and reserved 10,000 pairs as a test set for evaluation. This leaves us 866,195 pairs for training.

Recall that the random process involved in creating this dataset leads to pairs where positive examples are relatively easy to distinguish from negative ones. However, since we obtain the news articles through automatic content extraction, their text may have been extracted imperfectly (or may be entirely wrong in the case of outdated links). This makes training harder and guarantees imperfect scores during evaluation.

2) *Difficult & Small*: Working with the automatically retrieved dataset showed the need for harder training data. Models learned to achieve very high accuracy on the reserved test set but these results did not adequately reflect the perceived quality of the models. In fact, the need for hard examples to learn from and to evaluate on is not unique to our situation: The training in [10] is also directed to prefer difficult negative examples. Unlike there, we do not have a large amount of examples to choose the difficult ones from. If we arbitrarily select semantically close topics for the news articles from the *Easy & Large* dataset, we cannot be sure that these are legitimate negative examples. After all, there may be many relevant topics for a single news article.

Thus, we selected 65 news articles from the last year at random, provided semantically close topics and then asked 17 judges to manually label them as correct or incorrect. The semantically close topics were chosen as the union of the top 10 closest topics of various early versions of our models. This strategy is reminiscent of the pooling done in various competitions, e.g., [14].

Each judge could decide how many documents she or he wanted to annotate. Judges had the possibility to skip individual topics for the article they were reviewing. We only kept documents labeled by at least 3 judges and then only kept pairs for which there was a lead (either for *correct* or for *incorrect*) of at least 2. This leaves us 61 news articles with 180 positive and 2,362 negative topic-article pairs which we split into a training set with 2,042 pairs and a reserved evaluation set with 500 pairs.

B. Results

We compare the following setups, which are described in detail in Section IV: (Avg. GloVe) a decision boundary on the cosine between the average GloVe embeddings of the news article and descriptive documents; (KW Overlap) and (Entity Overlap) the MLP using the features to express overlap; (Pairwise) the MLP using the combination of the three previous setups; (RNN Enc.) a decision boundary on the L2 distance between the document encodings produced by our recurrent neural network; (Ensemble) the MLP operating on

the combination of the pairwise features and the L2 between document encodings; This ensemble was trained by using the encoder from the (RNN Encoding) setup and freezing its weights during training.

Table I depicts the performance of the models when trained only on the training set from our large, automatically-retrieved dataset. While the performance looks good on the corresponding test set, we clearly see how their performance is rather lackluster on the “hard cases”, i.e., the test set from Difficult & Small.

While we need the training set from the large dataset because of its sheer size, the manually labeled hard examples are actually a lot more valuable and can vastly increase the quality of our models. Table II shows the refined results when we include the training set from our Difficult & Small data into the model training data. We do so by combining both training sets and increasing the weight of the manual examples in the loss function by a factor of 2 over the automatically generated ones.

The results show that we lose very little to no quality on the Easy & Large dataset, but increase quality a lot on Difficult & Small. We are confident that additional manually tagged examples would enable our models to achieve even better performance. To substantiate this claim, we have trained the model (Ensemble) with all automatically retrieved training examples plus 0, 100, 500, 1000 and all 2042 manually tagged training examples.

The plots in Figure 4 show how “hard examples” to learn from can vastly improve the performance on the difficult dataset, whilst losing almost no quality on the easy one. In fact, for the model (Ensemble) examined in the figure, the performance on the easy dataset yields the exact same F1 values. Tables I and II provide more details here. The plots in Figure 4 indicate that further manually tagged examples would further improve the F1-Measure because the gradient of the Difficult & Small line did not decrease yet.

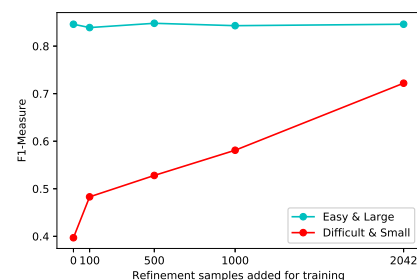


Fig. 4. Effect of adding n manually tagged examples for training the model (Ensemble).

VI. CONCLUSION & FUTURE WORK

We have presented and evaluated methods to detect topics in news articles by computing the similarity between the news

TABLE I
RESULTS ON THE RESERVED TEST SETS FROM EITHER DATASET WHEN TRAINING ONLY WITH THE AUTOMATICALLY RETRIEVED TRAINING EXAMPLES FROM EASY & LARGE.

	Easy & Large				Difficult & Small			
	Acc	Prec	Rec	F1	Acc	Prec	Rec	F1
Avg. GloVe	0.923	0.879	0.134	0.232	0.928	0.5	0.111	0.111
KW Overlap	0.949	0.763	0.603	0.677	0.786	0.141	0.389	0.207
Entity Overlap	0.956	0.777	0.704	0.739	0.796	0.234	0.806	0.363
Pairwise	0.964	0.841	0.728	0.78	0.842	0.258	0.639	0.368
RNN Enc.	0.958	0.746	0.784	0.765	0.746	0.199	0.833	0.321
Ensemble	0.973	0.84	0.852	0.846	0.812	0.258	0.861	0.397

TABLE II
RESULTS ON THE RESERVED TEST SETS FROM EITHER DATASET WHEN TRAINING ON THE COMBINATION OF TRAINING EXAMPLES.

	Easy & Large				Difficult & Small			
	Acc	Prec	Rec	F1	Acc	Prec	Rec	F1
Avg. GloVe	0.921	0.872	0.111	0.197	0.934	0.8	0.111	0.195
KW Overlap	0.949	0.764	0.605	0.676	0.794	0.139	0.361	0.202
Entity Overlap	0.959	0.816	0.637	0.716	0.832	0.250	0.667	0.364
Pairwise	0.966	0.819	0.780	0.799	0.820	0.255	0.778	0.384
RNN Enc.	0.976	0.894	0.821	0.856	0.960	0.735	0.694	0.714
Ensemble	0.978	0.889	0.808	0.846	0.960	0.722	0.722	0.722

article and representative topic documents. We have shown how two kinds of input can be used to compute effective measures for similarity: (1) simple numerical features to model the overlap of mentioned entities and keywords and (2) the full texts which are passed through a recurrent neural network to produce an encoding vector. Their combination outperforms each of them individually.

While our model architecture could still be improved, there are more important aspects that should be addressed in the immediate future. Our experiments have shown that the choice of training data plays a large role. In particular, manually labeled “hard cases” are very valuable.

The more manually labeled data we have, the better results we expect. Hence, collecting more such data should be an effective way to further improve our system.

Another big issue is the choice of available topics. We have argued how this choice could make the problem very easy or arbitrarily hard. Our approach is designed so that this choice can be made per use case, possibly in a manual fashion. The current heuristic (see Section III) already turns out to be a very nice starting point and thanks to its API, Wikipedia easily allows for hourly updates to the list of available topics. If topics obtained from Wikipedia see lots of usage, some form of (automated or even manual) curation would be very helpful, e.g., elimination of obscure topics and joining cases where one logical topic is arguably split across several Wikipedia articles.

On the technical side, the search for better word embeddings is probably more urgent than further experiments with model architectures. Combining the vectors of various kinds of pre-trained embeddings, operating on character-level input, and context-sensitive embeddings are all potentially very

valuable improvements. State-of-the-art systems for other NLP problems have demonstrated the value of these techniques, e.g., [15] for Named Entity Recognition. To take this idea even further, we want to experiment not only with pre-trained embeddings but entire pre-trained language models to refine and build upon. Inductive transfer learning from language models, e.g., ULMFit [16], has shown to be very powerful for multiple NLP tasks.

VII. ACKNOWLEDGMENTS

We want to thank Frank J. Balbach for a plethora of constructive comments.

REFERENCES

- [1] D. M. Blei, A. Y. Ng, and M. I. Jordan, “Latent dirichlet allocation,” *J. Mach. Learn. Res.*, vol. 3, pp. 993–1022, Mar. 2003.
- [2] T. Hofmann, “Probabilistic latent semantic indexing,” in *Proceedings of the 22Nd Annual International ACM SIGIR Conference on Research and Development in Information Retrieval*, ser. SIGIR ’99. New York, NY, USA: ACM, 1999, pp. 50–57.
- [3] D. Ramage, D. L. W. Hall, R. Nallapati, and C. D. Manning, “Labeled LDA: A supervised topic model for credit attribution in multi-labeled corpora,” in *Proceedings of the 2009 Conference on Empirical Methods in Natural Language Processing, EMNLP 2009, 6-7 August 2009, Singapore*, 2009, pp. 248–256.
- [4] T. Uslu, A. Mehler, A. Niekler, and D. Baumartz, “Towards a DDC-based topic network model of Wikipedia,” in *Conference: Proceedings of 2nd International Workshop on Modeling, Analysis, and Management of Social Networks and their Applications (SOCNET 2018)*, 2018.
- [5] A. Joulin, E. Grave, P. Bojanowski, and T. Mikolov, “Bag of tricks for efficient text classification,” *CoRR*, vol. abs/1607.01759, 2016.
- [6] J. Allan, J. G. Carbonell, G. Doddington, J. Yamron, and Y. Yang, “Topic detection and tracking pilot study final report,” in *Proceedings of the DARPA Broadcast News Transcription and Understanding Workshop*, 1998.

- [7] J. Allan, R. Papka, and V. Lavrenko, "On-line new event detection and tracking," in *SIGIR*, vol. 98. Citeseer, 1998, pp. 37–45.
- [8] H. Larochelle, D. Erhan, and Y. Bengio, "Zero-data learning of new tasks," in *Proceedings of the Twenty-Third AAAI Conference on Artificial Intelligence, AAAI 2008, Chicago, Illinois, USA, July 13-17, 2008*, 2008, pp. 646–651.
- [9] M. Palatucci, D. Pomerleau, G. E. Hinton, and T. M. Mitchell, "Zero-shot learning with semantic output codes," in *Advances in neural information processing systems*, 2009, pp. 1410–1418.
- [10] F. Schroff, D. Kalenichenko, and J. Philbin, "Facenet: A unified embedding for face recognition and clustering," in *Proceedings of the IEEE conference on computer vision and pattern recognition*, 2015, pp. 815–823.
- [11] J. Pennington, R. Socher, and C. D. Manning, "GloVe: Global vectors for word representation," in *Empirical Methods in Natural Language Processing (EMNLP)*, 2014, pp. 1532–1543.
- [12] T. Mikolov, I. Sutskever, K. Chen, G. S. Corrado, and J. Dean, "Distributed representations of words and phrases and their compositionality," in *Advances in neural information processing systems*, 2013, pp. 3111–3119.
- [13] Z. Yang, D. Yang, C. Dyer, X. He, A. Smola, and E. Hovy, "Hierarchical attention networks for document classification," in *Proceedings of the 2016 Conference of the North American Chapter of the Association for Computational Linguistics: Human Language Technologies*, 2016, pp. 1480–1489.
- [14] K. Balog, P. Serdyukov, and A. P. De Vries, "Overview of the trec 2010 entity track," Norwegian Univ of Science and Technology Trondheim, Tech. Rep., 2010.
- [15] A. Akbik, D. Blythe, and R. Vollgraf, "Contextual string embeddings for sequence labeling," in *COLING 2018, 27th International Conference on Computational Linguistics*, 2018, pp. 1638–1649.
- [16] J. Howard and S. Ruder, "Universal language model fine-tuning for text classification," in *Proceedings of the 56th Annual Meeting of the Association for Computational Linguistics, ACL 2018, Melbourne, Australia, July 15-20, 2018, Volume 1: Long Papers*, 2018, pp. 328–339.

Journal Information and Instructions for Authors

I. JOURNAL INFORMATION

Polibits is a half-yearly open-access research journal published since 1989 by the *Centro de Innovación y Desarrollo Tecnológico en Cómputo* (CIDETEC: Center of Innovation and Technological Development in Computing) of the *Instituto Politécnico Nacional* (IPN: National Polytechnic Institute), Mexico City, Mexico.

The journal has double-blind review procedure. It publishes papers in English and Spanish (with abstract in English). Publication has no cost for the authors.

A. Main Topics of Interest

The journal publishes research papers in all areas of computer science and computer engineering, with emphasis on applied research. The main topics of interest include, but are not limited to, the following:

- Artificial Intelligence
- Natural Language Processing
- Fuzzy Logic
- Computer Vision
- Multiagent Systems
- Bioinformatics
- Neural Networks
- Evolutionary Algorithms
- Knowledge Representation
- Expert Systems
- Intelligent Interfaces
- Multimedia and Virtual Reality
- Machine Learning
- Pattern Recognition
- Intelligent Tutoring Systems
- Semantic Web
- Robotics
- Geo-processing
- Database Systems
- Data Mining
- Software Engineering
- Web Design
- Compilers
- Formal Languages
- Operating Systems
- Distributed Systems
- Parallelism
- Real Time Systems
- Algorithm Theory
- Scientific Computing
- High-Performance Computing
- Networks and Connectivity
- Cryptography
- Informatics Security
- Digital Systems Design
- Digital Signal Processing
- Control Systems
- Virtual Instrumentation
- Computer Architectures

B. Indexing

The journal is listed in the list of excellence of the CONACYT (Mexican Ministry of Science) and indexed in the following international indices: Web of Science (via SciELO citation index), LatIndex, SciELO, Redalyc, Periódica, e-revistas, and Cabell's Directories.

There are currently only two Mexican computer science journals recognized by the CONACYT in its list of excellence, *Polibits* being one of them.

II. INSTRUCTIONS FOR AUTHORS

A. Submission

Papers ready for peer review are received through the Web submission system on www.easychair.org/conferences/?conf=polibits1; see also updated information on the web page of the journal, www.cidetec.ipn.mx/polibits.

The papers can be written in English or Spanish. In case of Spanish, author names, abstract, and keywords must be provided in both Spanish and English; in recent issues of the journal you can find examples of how they are formatted.

The papers should be structured in a way traditional for scientific paper. Only full papers are reviewed; abstracts are not considered as submissions. The review procedure is double-blind. Therefore, papers should be submitted without names and affiliations of the authors and without any other data that reveal the authors' identity.

For review, a PDF file is to be submitted. In case of acceptance, the authors will need to upload the source code of the paper, either Microsoft Word or LaTeX with all supplementary files necessary for compilation. Upon acceptance notification, the authors receive further instructions on uploading the camera-ready source files.

Papers can be submitted at any moment; if accepted, the paper will be scheduled for inclusion in one of forthcoming issues, according to availability and the size of backlog.

See more detailed information at the website of the journal.

B. Format

The papers should be submitted in the format of the IEEE Transactions 8x11 2-column format, see http://www.ieee.org/publications_standards/publications/authors/author_templates.html. (while the journal uses this format for submissions, it is in no way affiliated with, or endorsed by, IEEE). The actual publication format differs from the one mentioned above; the papers will be adjusted by the editorial team.

There is no specific page limit: we welcome both short and long papers, provided that the quality and novelty of the paper adequately justifies its length. Usually the papers are between 10 and 20 pages; much shorter papers often do not offer sufficient detail to justify publication.

The editors keep the right to copyedit or modify the format and style of the final version of the paper if necessary.

See more detailed information at the website of the journal.

