Disjoint Semi-supervised Spanish Verb Sense Disambiguation Using Word Embeddings

Cristian Cardellino
Facultad de Matemática, Astronomía, Física y Computación
Universidad Nacional de Córdoba
Córdoba, Argentina
ccardellino@famaf.unc.edu.ar

Laura Alonso i Alemany
Facultad de Matemática, Astronomía, Física y Computación
Universidad Nacional de Córdoba
Córdoba, Argentina
alemany@famaf.unc.edu.ar

***Abstract*—This work explores the use of word embeddings, also known as word vectors, trained on Spanish corpora, to use as features for Spanish verb sense disambiguation (VSD). This type of learning technique is named *disjoint semisupervised learning* [1]: an unsupervised algorithm is trained on unlabeled data separately as a first step, and then its results (i.e. the word embeddings) are fed to a supervised classifier. Throughout this paper we try to assert two hypothesis: (i) representations of training instances based on word embeddings improve the performance of supervised models for VSD, in contrast to more standard feature engineering techniques based on information taken from the training data; (ii) using word embeddings trained on a specific domain, in this case the same domain the labeled data is gathered from, has a positive impact on the model's performance, when compared to general domain's word embeddings. The performance of a model over the data is not only measured using standard metric techniques (e.g. accuracy or precision/recall) but also measuring the model tendency to overfit the available data by analyzing the learning curve. Measuring this overfitting tendency is important as there is a small amount of available data, thus we need to find models to generalize better the VSD problem. For the task we use SenSem [2], a corpus and lexicon of Spanish and Catalan disambiguated verbs, as our base resource for experimentation.**

I. Introduction

The traditional approach of WSD using supervised "hand-crafted features", i.e. features taken from the training data itself, specially the use of bag-of-words like features [3], carries some problems. E.g. since in this kind of representation most of the features occurs on a single instance basis, the instance vectors have high dimensionality, but little useful information (i.e. are sparse). The dimensionality adds to the computational cost of solving the problem with classifiers. This is specially critical with more complex classifiers (like neural networks).

As handcrafted features are so tightly related to the training data, it makes the representation fixed on the domain of the data. When dealing with new examples, if the features (e.g. bags-of-words, ngrams, etc.) weren't on the original training data, the examples may have little information present to be represented with the available features of the training data. Thus, handcrafted features do not adapt well to drastic domain changes.

In recent years the natural language processing community has been exploring other methods for word's representations (and use these to represent phrases, sentences, etc.). Besides the general objective of improving the performance from our supervised methods, the idea of using semisupervised training with the aid of unsupervised word representations is to deal with the problems stated in the above paragraphs. We want to deal with coverage (by having a larger pool of examples to train a model) as well as domain change (for tasks outside the domain of the main training resource), unsupervised word representations give a way to sort these problems out.

This work is structured in the following way: Section II introduces the related work done in the areas of word sense disambiguation (particularly verb sense disambiguation), as well as the usage of word embeddings for natural language processing tasks; Section III describes the resources used in the experimentation of this work; Section IV lays out the architecture followed in our experimentation; Section V shows the results obtained from the experimentation and does a general analysis on what we find out; finally, Section VI finalizes the work with general remarks of the experiments, the conclusions regarding hypotheses and establish the future work to be done.

II. Relevant Work

Ye and Baldwin [4], use selectional preferences extracted with semantic role labelling for verb sense disambiguation (VSD). Their VSD framework is based upon three components: extraction of disambiguation features, selection of the best disambiguation feature with respect to unknown data and the tuning of the machine learner's parameters. For their study they use a maximum entropy model. The VSD features they used include selectional preferences and syntactic features, e.g, bag-of-words, bag of PoS tags, bag of chunks; parsed tree based features using different levels of the tree as source of information; and non-parse trees based syntactic features, e.g., voice of the verb, quotatives, etc. This work is based on their

features for the comparison of word embeddings against a more traditional approach for VSD.

For Spanish there is little work done for VSD, and the work is more focused on general word sense disambiguation (WSD). Màrquez et al. [5] primarily focused on disambiguation of nouns. They used a three way approach: if the word has more than a threshold number of occurrences, it is classified with a support vector machines (SVM) classifier; if the word has less occurrences than the threshold it is assigned the most frequent sense (MFS) in the training corpus; if the word is not presented in the training corpus then it is assigned the MFS in WordNet. The SVM classifier features were a bag of words, n-grams of part-of-speech tags and lemmas, and syntactic label and syntactic function of the constituent that has the target noun as head. Other work in WSD with applications in Spanish is the work of Montoyo et al. [6] where the task consists in assigning the correct sense to words using an electronic dictionary as the source of word definitions. They present a knowledge-based method and a corpus-based method.

When it comes to using word embeddings, Turian et al. [7] improve the accuracy of different existing NLP systems by using unsupervised word representations as extra features. In their work, they evaluate three different unsupervised word representations: Brown clusters [8], Collobert and Weston [1] embedding, and HLBL embeddings of words [9]; and try them out on named entity recognition and chunking. Using these representations they effectively show improvement of performance in nearly state-of-the-art baselines. More recent years have seen the introduction to the *skip-gram model* and *continuous bag-of-words model* by Mikolov et al. [10]. These are novel model architectures for computing continuous vector representations of words from very large data sets. In particular, the skip-gram model is able to learn high-quality distributed vector representations that capture a large number of precise syntactic and semantic word relationships. The use of negative sampling [11] improve both the quality of the vectors and the training speed, by sub-sampling of the frequent words.

For WSD with word embeddings there is an evaluation study by Iacobacci et al. [12]. In this, they propose different methods through which word embeddings can be leveraged in a state-of-the-art supervised WSD system architecture, and perform analysis of how different parameters affect performance.

III. Resources

A. Labeled Corpus

SenSem [2] is a manually disambiguated corpus for verb in both Spanish and Catalan. It contains the 248 most common verbs of Spanish, annotated with senses defined in a provided lexicon, some of them with mappings to the Spanish WordNet Ontology [13].

A version of the SenSem corpus has part-of-speech tags automatically annotated with Freeling [14]. However these tags are annotated on a word based level, thus there is a large proportion of them annotated with the wrong tag (e.g. verbs annotated as nouns). Furthermore, Spanish has some words that are multi-words (i.e. words formed of two ore more different terms) which tag is not the same than those of each of the words compounding the multi-word. E.g. “más_allá_de” is tagged as a multi-word with Part-of-Speech tag “SP”, it’s a preposition, however, the words “más” and “allá’ are by themselves adverbs and only “de” is a preposition.

In order to gather information more useful for feature extraction, there was a two preprocessing steps of the SenSem corpus. First, an automatic annotation using a statistical dependency parser. In this step the SenSem’s sentences, which are tokenized, are parsed with Freeling’s statistical dependency parser. The sentences are automatically annotated with: lemma, part-of-speech tag, morphosyntactic information and dependency triples. Also, there is multi-word detection and named entity recognition (treated by Freeling as multi-words).

Nevertheless, the automatic annotation is not enough as errors come not only from Freeling but other problems SenSem has as well: sentences without a defined sense, sentences where the verb to disambiguate is not present and sentences truncated before finishing. For this reason, the second step of preprocessing was manual, where each of the automatically annotated sentences, where the main lemma to disambiguate was lost (because of mistagging, not being correctly marked in the original resource, etc.), is found manually. Besides this, all cases that are erroneous in the original corpus (e.g. truncated sentences or sentences without a defined sense) were discarded.

After the preprocessing step, the SenSem corpus was split in train/test. For this all those senses with only one occurrence in the corpus are filtered out and the remaining senses are split with stratified sampling using 80% for training and 20% for testing, where the training and the test corpus have each at least one occurrence of every sense and at most a percentage of samples of each sense similar to the complete set. This was done in order to have feedback in the test set regarding those classes appearing the least number of times, which is different to the circumstances in which these kind of systems work in real environments; thus, the experimental results which follow should be understood as the best we are able to obtain in some of the most favorable conditions.

Table I shows some of the statistics of the SenSem corpus after the preprocessing of the text and removal of erroneous sentences[1].

B. Word embeddings

This work focus on the embeddings obtained with Word2Vec algorithm [10]. The original word embeddings used for the experiments are the pre-trained Spanish Billion Word Corpus and Embeddings (SBWCE) [15].

The SBWCE corpus is a compilation of nearly 1.5 billion words of Spanish language taken from different sources available on the Internet, most of them coming from corpus used

[1]The final resource is available at: https://cs.famaf.unc.edu.ar/~ccardellino/resources/grial/sensem.conll.bz2

Statistic	Value
No. of sentences	24153
No. of sentences filtered out	219
No. of sentences for training	19071
No. of sentences for testing	4863
Average no. of sentences per lemma (after filter)	96.51
Median no. of sentences per lemma (after filter)	98.00
Average no. of sentences per sense (after filter)	31.00
Median no. of sentences per sense (after filter)	16.00
Average no. of sentences per lemma for training	76.90
Average no. of sentences per lemma for test	19.61
Average no. of sentences per sense for training	24.70
Average no. of sentences per sense for test	6.30
No. of lemmas (after filtering)	110
No. of senses (after filtering)	421
Average no. of senses per lemma	3.82
Median no. of senses per lemma	3.00

Table I
SENSEM STATISTICS

for statistical machine translation tasks, as well as corpus from the Wikimedia foundation, making it a heterogeneous domain corpus.

The word embeddings pre-trained from this resource was created using Word2Vec's *skip-gram* model and the gensim library [16]. It filters out words with less than 5 occurrences leaving out roughly 1 million unique words. The final word vectors dimension is 300.

The general idea of using pre-trained embeddings is the availability of them. In general terms, embeddings trained on big amount of data perform relatively well for general tasks, however, we wanted to see the impact of training embeddings specifically for the data available and what effect does this have on the results.

SenSem provides an annotated corpus based on a small fraction of two newspapers from the region of Catalunya in Spain: "El Periódico" and "La Vanguardia". This make the resource heavily based on senses which have more to do with the journalistic domain. We trained word embeddings based on journalistic sources available on the SBWCE and other newspapers available online, particularly the corpora provided by the two newspapers on which SenSem is based on[2].

In comparison to the SBWCE corpus, the corpus we could gather for this task was much smaller, having nearly 71 million words available, which became 70 million after filtering out all those words with less than 3 occurrences. There was a final list of approximately 240 thousand unique words to generate the word embeddings which dimension was 50.

IV. EXPERIMENTAL SETTING

A. Basic layout

The verb sense disambiguation task is done per lemma. This means that we do not train one classifier for all the different senses, rather one classifier per each lemma with more than 1 sense available (and we omit those cases in the labeled corpus of lemmas with only 1 sense). There is not a single model to classify all the instances, but there are multiple models for each of the lemmas available. Having many models, we decided to visualize the results using boxplots.

[2]The resource is available at https://cs.famaf.unc.edu.ar/~ccardellino/resources/word_vectors/journal.wordvectors.bin.gz

B. Feature engineering

1) Supervised handcrafted features: As one of the goals of this work is to asses the impact of word embeddings, an unsupervised form of feature engineering, in the VSD task, we need to compare our results against a traditional approach using handcrafted features.

As explained in Section II, the most useful information for WSD seems to be word forms or lemmas co-occurring with a given sense, alongside syntactic information [17]. The selected features were originally based on the work by Cardellino [18]. However, we proceeded to do some further exploration of features, following the lines of Ye and Baldwin [4]. After some experimentation we finally decided to go further with the following features:

- The main word.
- The main word's lemma.
- The main word's part-of-speech tag: in the case of Spanish part-of-speech tags, only the abbreviated form is used (generally the 2 or 3 first letters).
- In case of Spanish, the morphosyntactic information of the main word is given separately from the part-of-speech tag.
- The bag-of-words of a symmetric 5-word window (i.e. 5 words before and 5 words after the main word): this feature represents the number of occurrences of each words surrounding the main word (without considering it) giving no importance to the position.
- The words, lemmas and part-of-speech tags of the surrounding words in a 5-word window at the corresponding position.
- The bigram and trigram formed by the words before and after the main word.
- The dependency triples formed by the main word, the relation and the words dependant on the main word (inbound dependency triples). And the dependency triple formed by the main word, the relation and the word from which the main word depends or if it is the root word (outbound dependency triple).

However, there is a problem with these features. As the dimensionality of the data was too much to handle for the multilayer perceptron classifier, we explored two ways to reduce the dimensionality: *univariate feature selection* using the ANOVA F-value [19], as well as the *hashing trick* [20]. We first compare the results using classifiers that support high dimension of features, e.g. SVM or Decision Trees, and saw there was not a visible difference between using any kind of representation, thus decided to use the hashing trick as it provided a good representation, that was more efficient than having to run feature selection over the dataset before every experiment.

2) Word embeddings features: The word embeddings are straightforward to use. The idea is to represent each instance (i.e. the sentence with the verb to disambiguate) as a concatenation of word vectors. We use the token of the verb to disambiguate as the central vector in the concatenation, and chose a symmetric window of 5 tokens at each side of the central word making the final vector a concatenation of 11 words.

If the token is not available in the word embeddings model we try the token with all lowercase characters and capitalized (first character uppercase and the rest lowercase). If neither version of the token is available we use a vector of zeros of the same dimension that the word embeddings.

For the case when the central word is near to the beginning or end of the sentence, we pad the amount of words left to complete the whole vector with zeros. E.g., the verb is located as the third word from the beginning of the sentence, then to complete the right window we use the word vectors for the first and second token of the sentence and pad with three zero valued vectors before the vectors of two tokens.

Following this adjustment, the input vector when using the SBWCE corpus are of dimension 3300 and the vectors for the journalistic domain are of dimension 550.

C. Performance measure

Word sense disambiguation has a Zipfian [21] distribution over it's data. Thus, there are certain metrics which can affect the perception of how good or bad an algorithm is, because they show better results only when the most frequent class shows better results. Accuracy is a classic example of a biased metric, it measures the percentage of correct guesses from an algorithm, and if the most frequent class shows a large proportion of examples over the whole dataset, accuracy shows a good result, even for a simple baseline (e.g the algorithm that maps every instance to the most frequent class). In this work we rely on other metrics. In particular, precision and recall are good metrics but they are class-based, which in cases like this, with large amount of lemmas, each having many classes, results become difficult to follow. We appeal to the use of two averages for precision and recall: *macro average*, and *weighted average*.

The first one is defined as the unweighted mean of the values for each sense [22]. In this metric the least frequent senses are as important as the most frequent ones, nevertheless, it also means that extreme class imbalance will drastically reduce the final results. Weighted average is calculated by averaging the metric of each class weighted by the number of occurrences it has. Classes with more occurrences have more relevance in the final results.

These five metrics give a better idea of the model's performance, whether if is really improving or only showing better results for the most frequent classes.

D. Tendency to overfit

Another important measure in this work is the tendency to overfit a model has when new examples are added. This can be measured by analyzing the learning curve of an algorithm and watching over the *error due to high variance*. This error is the amount by which the prediction, over one training set, differs from the expected predicted value, over all the training sets. Manning et al. [22] defines it as the variation of the prediction of learned classifiers: it measures how inconsistent are the predictions from one another, over different training sets, not whether they are accurate or not. The learning curve is calculated with the following algorithm:

1) Split the whole corpus evenly in n parts.
2) Take the first part and split it again using stratified k-fold cross-validation.
3) Take $k-1$ folds, train a model and test it against the fold that was left out, saving the error (in this case, cross entropy) for both the training data and test data.
4) Repeat the previous step for each k fold.
5) Add another part to the dataset and repeat steps (3) and (4) until all the data is added.
6) Show the mean and variance of the error of the training and test sets for each step of the algorithm.

The resulting curve shows the variance error of a model as the number of training examples increases. Models with higher variance are more prone to overfit the data.

Metric	DT	MLP	NB	LR	SVM
Accuracy mean	0.72	0.74	0.72	0.76	0.77
Precision macro avg mean	0.48	0.47	0.35	0.46	0.49
Precision weighted avg mean	0.69	0.70	0.61	0.69	0.71
Recall macro avg mean	0.49	0.48	0.38	0.46	0.49
Recall weighted avg mean	0.72	0.74	0.73	0.76	0.77

Table II
PERFORMANCE METRICS FOR SUPERVISED CLASSIFIERS: DECISION TREE (DT), MULTILAYER PERCEPTRON (MLP), NAIVE BAYES (NB), LOGISTIC REGRESSION (LR) AND SUPPORT VECTOR MACHINES (SVM)

V. RESULTS AND ANALYSIS

A. Classifier selection

We wanted to see the impact of the word embeddings both in linear and non-linear classifiers. Based on the work by Cardellino [18], we initially checked both naive Bayes and support vector machines. However, we also explored logistic regression as another possible linear algorithm, and finally for non-linear algorithms we went with decision trees and multilayer perceptron with one hidden layer. In general terms, only naive Bayes showed visible worse results, the other 4 methods had a similar performance. To keep it simple for this work, we decided to pick only one linear classifier and only one non-linear to see how the different features affected the two kind of classifiers. Table II shows the mean of the models for each of the selected metrics according to Section IV-C.

For choosing a non-linear classifier, the selected method was **multilayer perceptron**. Although decision tree had a similar performance when looking at the results in the Table, it suffered from *error due to high variance*, as measured by the metric defined on Section IV-D.

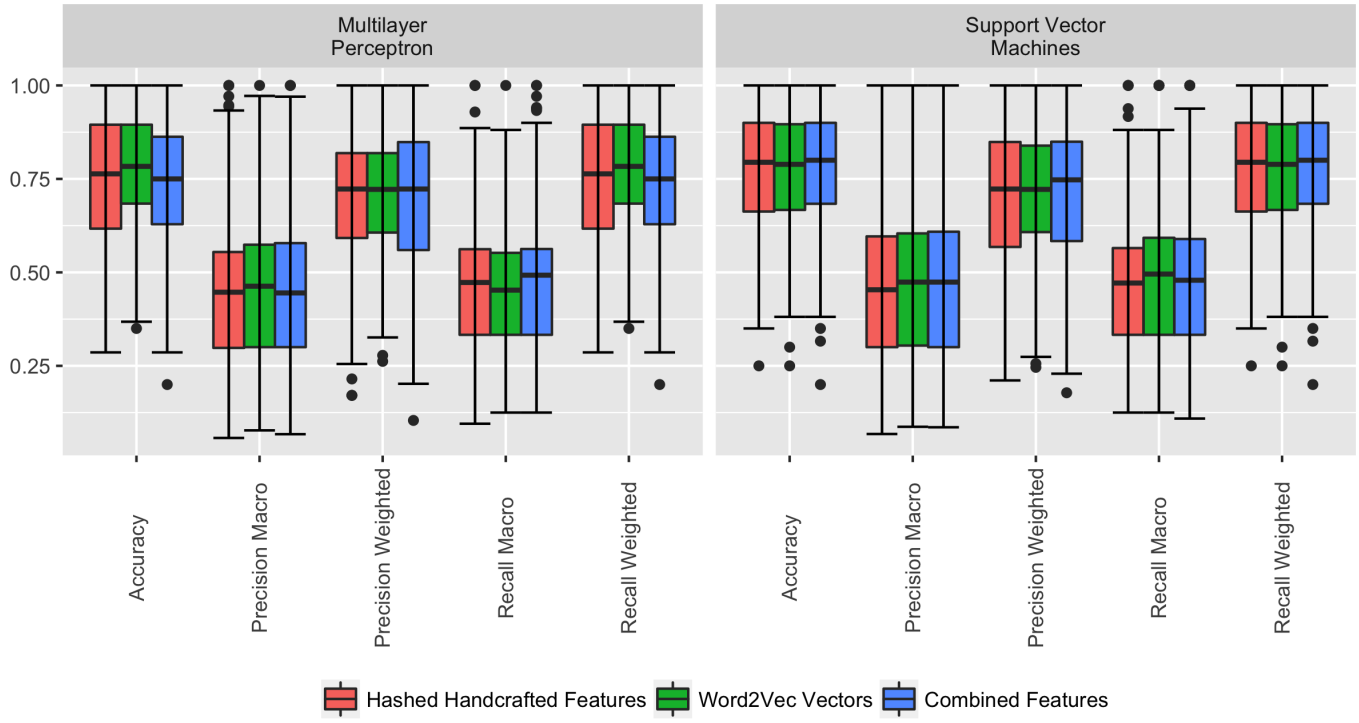

Figure 1. Performance comparison of handcrafted features, word embeddings and combined features.

For linear classifiers, we choose **support vector machines** as it showed slightly better results than the logistic regression classifier, however those results were not statistically significant. The decision of using it was just for simplicity in this work, which is not concerned so much in the specific performance of the model rather in how it is affected by the use of word embeddings.

B. Comparison of word embeddings and handcrafted features

The first experiment compares the task of verb sense disambiguation (VSD) using handcrafted features and using word embeddings features. We also explored the combination of handcrafted features alongside word embeddings. The goal is to see the performance using different representation of the instances.

Figure 1 shows the performance measured by metrics of Section IV-C: accuracy, precision macro and weighted average, and recall macro and weighted average. It uses a box and whiskers plot to visualize the performance for all the lemmas. The boxplots show the median (the black line inside the box) and the quartiles (the beginning and end of each box and the whiskers), of the performance for all the models (one per each lemma). As explained in Section V-A, we only show the performances for the multilayer perceptron (MLP) classifier and support vector machines (SVM).

Word embeddings have slightly better performance (the boxplots are higher) than handcrafted features for almost all metrics. With the combination of features is a little harder to figure out if there's a real improvement or not, it is not clearly visible over word embeddings. In particular, for MLP, in some cases the performance suffers in certain part.

Figure 2 shows the difference between representations. This time we plot the learning curve of the classifier for 3 splits (33, 66 and 100 percent of the training data) and 3 folds. The lines show the mean of the error for all the models (one per lemma), and the ribbons show the variance of the error for the sets obtained as the metric explained in Section IV-D states.

It starts off with the progress of the training dataset for each classifier. MLP shows a low error (almost close to zero) from the start, with little variance for it, i.e. it fits the training data almost flawlessly. For SVM is harder to adapt to the training data and this is a consequence of dealing with a linear classifier in a problem that seems to be non-linear. It eventually reaches a low error with almost no variance for the training data but never as low as MLP.

MLP has worse results when combining the features, consequence of them belonging to different spaces. It needs more fine tuning to reach better results, e.g. some type of normalization to reduce the high variance suffered when combining features. However, MLP is the classifier to have a more positive impact when dealing with word embeddings. Here we can hypothesize: the small amount of training data per lemma and the high sparsity of handcrafted features makes it easy for the MLP model to find a mapping between each training instance and the respective class (in this case sense) in the training data. This doesn't happen when using word vectors

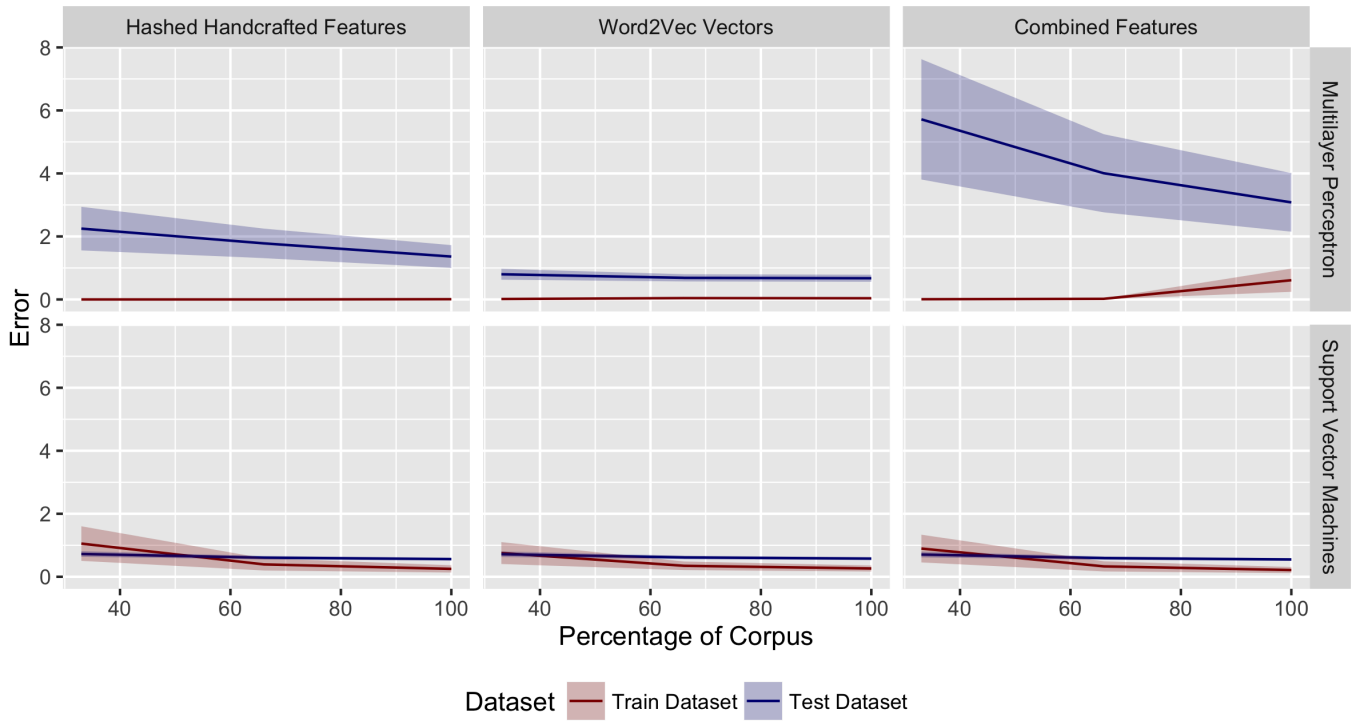

Figure 2. Learning curve comparison of handcrafted features, word embeddings and combined features.

because the dense representation makes the possibilities more difficult to map directly without overlapping them. It would need perhaps a deeper study on different architectures and perhaps other ways to deal with highly sparse representations.

SVM is more resilient to be affected by the extra data even if it comes from different spaces, like in the combination of handcrafted and word embeddings features. One thing to notice easily is the uniformity of the test error in all the iterations. Though it shows being better than what MLP offers (except when using only word embeddings as the test error has little difference for both classifiers there), the uniformity can be solely because the model heavily relays on separating the hyper-planes for the two (or maybe three) more frequent classes per lemma thus providing a low error in general.

In any case SVM also experiences a boost in performance, albeit not as visible as the case of MLP, when it comes to using word embeddings, in comparison of handcrafted features, which supports our hypothesis of word embeddings being better for this cases of small data than features engineered for the data itself.

C. Word embeddings domain comparison

As we saw in previous Section, word embeddings have a positive impact on the task of VSD. Now, we explore the domain of the embeddings and how this impacts on performance. As explained in Section III-B, we trained embeddings specific for the journalistic domain. We proceeded to test this against the same experiments described in previous sections which compares word embeddings with handcrafted features.

Our first experiments on this subject showed a great improvement of both classifiers when dealing with the journalistic domain embeddings, in comparison to the pre-trained SBWCE embeddings. This boosted improvement raise our attention. Since the journalistic corpus is smaller both in total number of words as well as dimension of embeddings, we wanted to rule out the possibility of the boost in performance being a consequence of a large corpus like SBWCE having a lot of noise. Particularly, SBWCE is based in large part on a Wikipedia dump, which is a source having not only paragraphs but also lists, tables, and other elements which can end up affecting the Word2Vec algorithm. For this we trained another set of word embeddings based on a random subsampling of the SBWCE corpus. This subsampling had around 70 million words, to follow the pattern of the journalistic domain corpus, and trained embeddings with the same characteristics as the journalistic domain corpus regarding minimum word count and embeddings dimension.

Figure 3 shows the performance results for the different embeddings domains: the SBWCE pre-trained embeddings, the embeddings trained from a random sample of the SBWCE, and finally the embeddings trained with the journalistic domain. It strikes that effectively doing a sample over the SBWCE pre-trained embeddings did some improvement, which confirmed our suspicions on the SBWCE embeddings being susceptible to the noise of the large corpus itself, however, in any case

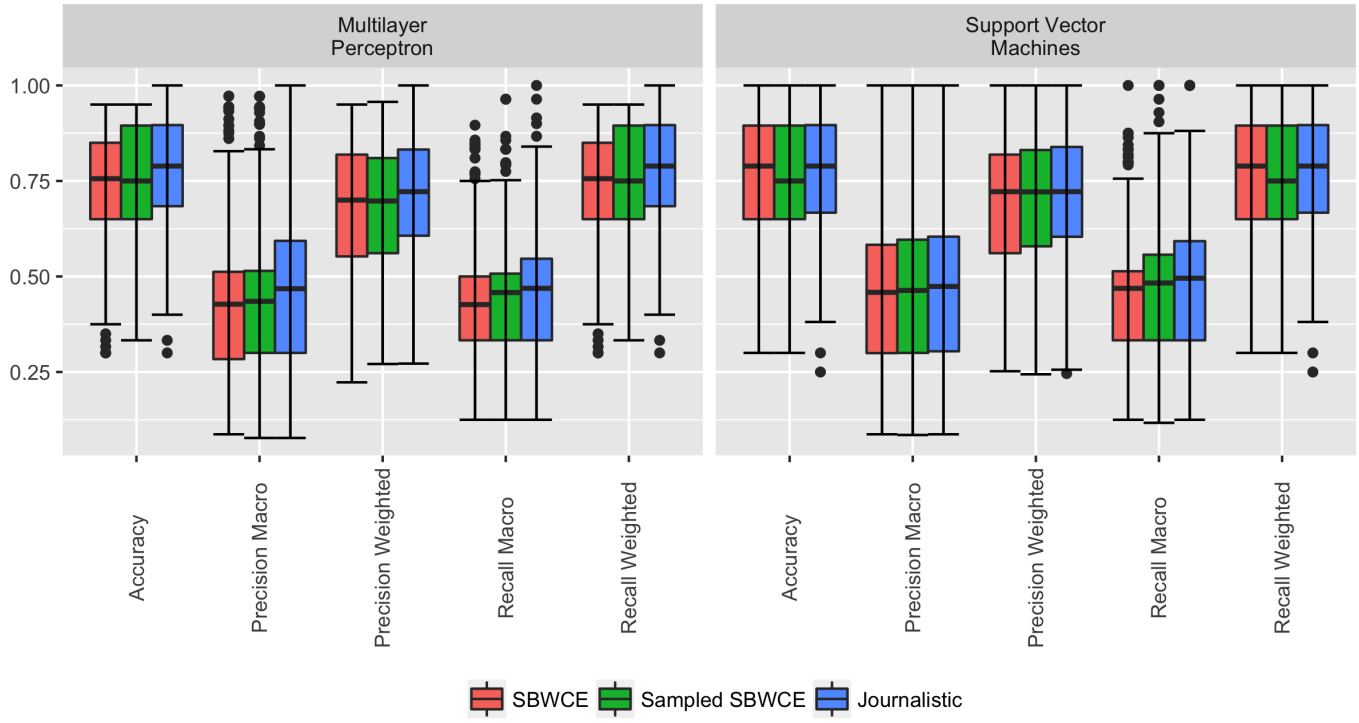

Figure 3. Performance comparison of word vectors domains.

the journalistic corpus is still the one having the better performance for the task. This performance boost is again more visible for the MLP classifier, however the SVM classifier still shows an improvement.

Figure 4 shows the learning curve following the experiments described in Section IV-D. It does so again for the three embeddings domains and the two classifiers. Again MLP classifier is more susceptible to change in domain and in general the noise. Is easy to see the difference between the full and the sampled version of SBWCE embeddings. SVM, on the other hand is more resistant to noise, although still has an improvement by removing it from the embeddings.

Something curious to notice is how the journalistic corpus seems to slightly improve the results for the MLP classifier but does not do so for the SVM classifier, rather has a slightly higher variance when dealing with training set error. Still, the difference in results is so little it can be just a matter of chance rather than a significant result. It does require some further investigation which is left for future work. In any case, the change in domain affects the performance in a visible way, but when measuring tendency to overfit, it does not seem to have a great impact on it, rather the use of a smaller corpus not subject of much noise is what improves the tendency to overfit of the model. What Figure 4 shows better, which is not so easily appreciated in Figure 2, is the difference between classifiers, as MLP does have a slightly higher error in the test set (and more variance) than SVM. In the same way, SVM cannot perfectly fit the training data like MLP does, which gives us more reasons to believe the problem is non-linear. In any case, we have to keep in mind that the number of training elements here is reduced, but for both classifiers incrementing the amount of elements reduces the general tendency to overfit of the models.

VI. Conclusions and Future Work

We studied how word embeddings impact on verb sense disambiguation for Spanish. We started off from two hypotheses: word embeddings effectively improve results over purely supervised feature engineering; and the domain on which the embeddings are created determines the final performance of a model.

To measure the performance we were interested not only in standard metrics like accuracy, precision and recall, but we also wanted to asses the impact of the different features in terms of tendency to overfit the data. This is why we measured the error due to variance in different steps of the classifier learning.

We showed promising results to support our hypotheses, as we effectively reached an improvement in performance using word embeddings, and moreover an improvement in the error due to variance.

There are still aspects we left out needing further revision and are subject of future work. First, we should try with some other kind of unsupervised representation, like the ones listed by Turian et al. [7]: Collobert and Weston [1] and Brown clusters [8].

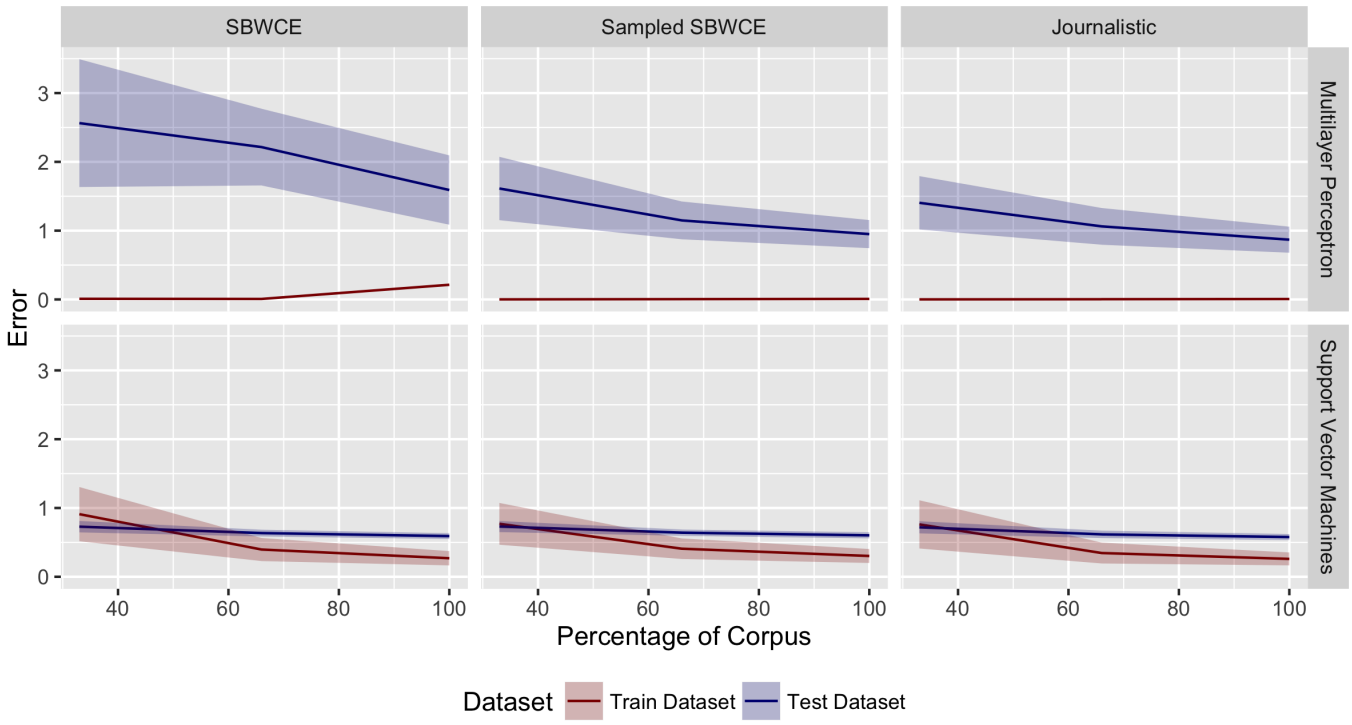

Figure 4. Learning curve comparison of word vectors domains.

Another line of work would be doing a more thorough error analysis on the complete SBWCE corpus, seeing if a better preprocessing of the data can provide better results. Finally, SVM also needs a further error analysis based on the results showed regarding high error variance in the training data in contrast to what MLP showed.

References

[1] J. Weston, F. Ratle, and R. Collobert, "Deep learning via semi-supervised embedding," in *Proceedings of the 25th International Conference on Machine Learning*, ser. ICML '08. New York, NY, USA: ACM, 2008, pp. 1168–1175. [Online]. Available: http://doi.acm.org/10.1145/1390156.1390303

[2] L. Alonso, J. Capilla, I. Castellón, A. Fernández, and G. Vázquez, "The sensem project: Syntactico-semantic annotation of sentences in spanish." in *Selected papers from RANLP 2005*, N. N. et al., Ed. John Benjamins, 2007, pp. 89–98.

[3] N. Ide and J. Véronis, "Introduction to the special issue on word sense disambiguation: The state of the art," *Comput. Linguist.*, vol. 24, no. 1, pp. 2–40, Mar. 1998. [Online]. Available: http://dl.acm.org/citation.cfm?id=972719.972721

[4] P. Ye and T. Baldwin, "Verb sense disambiguation using selectional preferences extracted with a state-of-the-art semantic role labeler," in *Proceedings of the Australasian Language Technology Workshop 2006*, Sydney, Australia, November 2006, pp. 139–148.

[5] L. Màrquez, L. Padró, M. Surdeanu, and L. Villarejo, "UPC: Experiments with Joint Learning within SemEval Task 9," in *Proceedings of the 4th International Workshop on Semantic Evaluations (SemEval-2007)*, 2007.

[6] A. Montoyo, M. Palomar, G. Rigau, and A. Suárez, "Combining knowledge- and corpus-based word-sense-disambiguation methods," *CoRR*, 2011.

[7] J. Turian, L. Ratinov, and Y. Bengio, "Word representations: A simple and general method for semi-supervised learning," in *Proceedings of the 48th Annual Meeting of the Association for Computational Linguistics*, ser. ACL '10. Stroudsburg, PA, USA: Association for Computational Linguistics, 2010, pp. 384–394. [Online]. Available: http://dl.acm.org/citation.cfm?id=1858681.1858721

[8] P. F. Brown, P. V. deSouza, R. L. Mercer, V. J. D. Pietra, and J. C. Lai, "Class-based n-gram models of natural language," *Comput. Linguist.*, vol. 18, no. 4, pp. 467–479, Dec. 1992. [Online]. Available: http://dl.acm.org/citation.cfm?id=176313.176316

[9] A. Mnih and G. E. Hinton, "A scalable hierarchical distributed language model," in *Advances in Neural Information Processing Systems 21*, D. Koller, D. Schuurmans, Y. Bengio, and L. Bottou, Eds. Curran Associates, Inc., 2009, pp. 1081–1088. [Online]. Available: http://papers.nips.cc/paper/3583-a-scalable-hierarchical-distributed-language-model.pdf

[10] T. Mikolov, K. Chen, G. Corrado, and J. Dean, "Efficient Estimation of Word Representations in Vector Space," Jan. 2013.

[11] T. Mikolov, I. Sutskever, K. Chen, G. Corrado, and J. Dean, "Distributed Representations of Words and Phrases and their Compositionality," Oct. 2013.

[12] I. Iacobacci, M. T. Pilehvar, and R. Navigli, "Embeddings for word sense disambiguation: An evaluation study," in *Proceedings of the 54th Annual Meeting of the Association for Computational Linguistics (Volume 1: Long Papers)*. Berlin, Germany: Association for Computational Linguistics, August 2016, pp. 897–907. [Online]. Available: http://www.aclweb.org/anthology/P16-1085

[13] A. Fernández-montraveta, A. Fernández-montraveta, G. Vázquez, and C. Fellbaum, "The spanish version of wordnet 3.0," *TEXT RESOURCES AND LEXICAL KNOWLEDGE*, pp. 175–182, 2008.

[14] L. Padró and E. Stanilovsky, "Freeling 3.0: Towards wider multilinguality," in *Proceedings of the Language Resources and Evaluation Conference (LREC 2012)*. Istanbul, Turkey: ELRA, May 2012.

[15] C. Cardellino, "Spanish Billion Words Corpus and Embeddings," http://crscardellino.me/SBWCE/, March 2016.

[16] R. Řehůřek and P. Sojka, "Software Framework for Topic Modelling with Large Corpora," in *Proceedings of the LREC 2010 Workshop on New Challenges for NLP Frameworks*. Valletta, Malta: ELRA, May 2010, pp. 45–50, http://is.muni.cz/publication/884893/en.

[17] S. Pradhan, E. Loper, D. Dligach, and M. Palmer, "Semeval-2007 task-17: English lexical sample, srl and all words," in *Proceedings of*

the Fourth International Workshop on Semantic Evaluations (SemEval-2007), 2007.

[18] C. Cardellino, "A Study of Semi-Supervised Spanish Verb Sense Disambiguation," in *Proceedings of the ESSLLI 2015 Student Session*, Barcelona, Spain, 2015, pp. 175–186.

[19] R. Fisher, "On the probable error of a coefficient of correlation deduced from a small sample," *Metron*, vol. 1, pp. 3–32, 1921.

[20] K. Weinberger, A. Dasgupta, J. Attenberg, J. Langford, and A. Smola, "Feature hashing for large scale multitask learning," 02 2009. [Online]. Available: https://arxiv.org/abs/0902.2206

[21] G. Zipf, *Human Behavior and the Principle of Least Effort*. Cambridge, MA: Addison–Wesley, 1949.

[22] C. D. Manning, P. Raghavan, and H. Schütze, *Introduction to Information Retrieval*. New York, NY, USA: Cambridge University Press, 2008.