HHU at SemEval-2017 Task 2: Fast Hash-Based Embeddings for Semantic Word Similarity Assessment

Behrang Qasemizadeh
DFG SFB 991
Universität Düsseldorf
Düsseldorf, Germany
zadeh@phil.hhu.de

Laura Kallmeyer
DFG SFB 991
Universität Düsseldorf
Düsseldorf, Germany
kallmeyer@phil.hhu.de

Abstract

This paper describes the HHU system that participated in Task 2 of SemEval 2017, Multilingual and Cross-lingual Semantic Word Similarity. We introduce our unsupervised embedding learning technique and describe how it was employed and configured to address the problems of monolingual and multilingual word similarity measurement. This paper reports from empirical evaluations on the benchmark provided by the task's organizers.

1 Introduction

The goal of Task 2 of SemEval-2017 is to provide a reliable benchmark for the evaluation of monolingual and multilingual semantic representations (Camacho-Collados et al., 2017). The proposed evaluation benchmark goes beyond classic semantic relatedness tests by providing both monolingual and cross-lingual data sets that include multiword expressions, domain-specific terms, and named entities for five languages. To measure 'semantic similarity' between pairs of lexical items, the HHU system uses the algorithm proposed in (QasemiZadeh et al., 2017), which is based on a derandomization of the 'random positive-only projections' method proposed by QasemiZadeh and Kallmeyer (2016).

Word embedding techniques (i.e., using distributional frequencies to produce word vectors of reduced dimensionality) are one of the most popular approaches to semantic word similarity problems. These methods are often rationalized using Harris' Distributional Hypothesis that words of similar linguistic properties appear with/within a similar set of 'contexts' (Harris, 1954). For example, words of related meanings co-occur with similar context words $\{c_1, \dots c_n\}$. This hypothesis implies that if these context words are grouped randomly into m buckets, e.g. $\{\{c_1 \dots c_x\}_1, \dots, \{c_y, \dots c_n\}_m\}$, then co-related words still co-occur with similar sets of buckets. QasemiZadeh and Kallmeyer (2016) exploit this assumption and propose random positive-only projections for building word vectors directly at a reduced dimensionality m. In this paper, we propose a derandomization of this method and a hash-based technique for learning word embeddings. In Section 2, we describe our method. In Section 3, we report results obtained by applying this method to the shared-task benchmark. Finally, we conclude in Section 4.

2 Method

Our method consists of two logical routines: (a) a *text skimmer* to collect co-occurrence information; and (b) a *hash-based encoder* to build low-dimensional vectors from collected co-occurrences in (a). Evidently, these procedures can be merged and ordered differently to meet requirements of an application.

To build an m-dimensional embedding for an entity w (such as a word or phrase) that co-occurs with (or within) some context elements c (resulting from the skimming routine), we take the following steps:

Algorithm 1 : Encoding Co-Occurrences

1: $\vec{w} = \vec{0}$
2: **for each** c co-occurring with w **do**
3: $\quad \mathtt{d} \leftarrow \mathtt{abs}(\mathtt{hash}(c)\ \%\ m)$
4: $\quad \vec{w}_{\mathtt{d}} = \vec{w}_{\mathtt{d}} + 1$
return $\vec{w}$

Here, $w_{\mathtt{d}}$ is the dth component of $\vec{w}$. The `hash` function assigns a hash code (e.g., an integer) to each context element c. The `abs` function returns the absolute value of its input number and `%` is the

Proceedings of the 11th International Workshop on Semantic Evaluations (SemEval-2017), pages 250–255,
Vancouver, Canada, August 3 - 4, 2017. ©2017 Association for Computational Linguistics

modulus operator and it gives the remainder of the division of the generated hash code by the chosen value m. We use the following `hash` function:[1]

```
int hash(byte[] key) {
  int i = 0;
  int hash = 0;
  while (i != key.length) {
    hash += key[i++];
    hash += hash << 10;
    hash ^= hash >> 6;
  }
  hash += hash << 3;
  hash ^= hash >> 11;
  hash += hash << 15;
  return hash;
}
```

Our choice for `hash` is motivated by its low collision rate for short words (byte sequences) and the closer resemblance of computed ds to an independent and identical distribution (i.i.d). It can be verified that the procedure proposed above implements a derandomization of QasemiZadeh and Kallmeyer's POP method: The generated modulus of hash codes from context elements constitutes a random positive-only projection matrix, and the component-wise additions compute the multiplication of this randomly generated matrix with the original high-dimensional vectors (QasemiZadeh et al., 2017).

2.1 Computing Similarities

Once $\vec{w}$s are constructed, they are weighted by the expected and marginal frequencies, e.g., using positive pointwise mutual information (PPMI) (Church and Hanks, 1990; Turney, 2001). Let $\mathbf{W}_{p\times m}$ (consisting of p row vectors $\vec{w}$ of dimensionality m) be the set of embeddings in our model (i.e., the output of Algorithm 1). The PPMI weight for a component w_{xy} in $\mathbf{W}$ is given by:

$$ppmi(w_{xy}) = \max(0, \log \frac{w_{xy} \times \sum_{i=1}^{p}\sum_{j=1}^{m} w_{ij}}{\sum_{i=1}^{p} w_{iy} \times \sum_{j=1}^{m} w_{xj}}).$$

For this task, however, we adopt *cascaded PPMI weightings*: PPMI-weighted vectors are weighted once more using the above-mentioned formula, i.e., we compute $ppmi(ppmi(\mathbf{W}_{p\times m}))$. We believe this cascaded weighting yields better results by providing a well-balanced scaling of the original PPMI weights. Note that the weighting process is fast since it is carried out on vectors of small dimensionality m.

[1]Designed by Bob Jenkins (2006); see http://www.burtleburtle.net/bob/hash/doobs.html.

Finally, we compute similarities between these weighted vectors using a correlation measure. QasemiZadeh and Kallmeyer (2016) suggest Pearson's r for PPMI weighted vectors. Later, in QasemiZadeh et al. (2017), they suggest Goodman and Kruskal's γ coefficient (Goodman and Kruskal, 1954). To compute γ, *concordant* and *discordant* pairs must be counted. Given any pairs such as (x_i, y_i) and (x_j, y_j) from two m-dimensional vectors $\vec{x}$ and $\vec{y}$ and the value $v = (x_i - x_j)(y_i - y_j)$, these two pairs are concordant if $v > 0$ and discordant if $v < 0$. If $v = 0$, the pair is neither concordant nor discordant. Let p and q be the number of concordant and discordant pairs, then γ is given by (Chen and Popovich, 2002, p. 86):

$$\gamma = \frac{p - q}{p + q}.$$

In this paper, we suggest a new estimator based on Lin's information theoretic definition of similarity (Lin, 1998):

$$sim_{lin} = \log(\frac{2\times\sum_{i=1}^{m}(x_i y_i)(1+\log(2+x_i y_i))}{\sum_{i=1}^{m} x_i^2 + \sum_{i=1}^{m} y_i^2}).$$

2.2 Extending the Method to Cross-Lingual Tasks

The proposed method can also be employed in a cross-lingual setting. However, this requires a small dictionary (translation-memory) and an additional pre-processing step.

In the pre-processing step, all pairs of lexical items in the input dictionary must be first mapped onto a common symbol space. Let's assume that the input dictionary consists of entries of the form $l \mapsto \{t_1, \ldots, t_n\}$ (i.e., l is a lexical item in the source language which has a number of t_i translations in the target language). To build the common symbol space, we generate all possible (l, t_i) tuples and we assign them *unique* identifiers—i.e., $(l, t_i) \mapsto s$. Finally, these tuples and their assigned identifiers are flattened in a symbol table t: for instance, if (l, t_i) are assigned to the unique identifier s, then the entries of (l, s) and (t_i, s) are stored in this table t. Note that the mappings in t are not necessarily one-to-one.

To build cross-lingual vectors for lexical items w in any of the input languages, similar to the monolingual setting, input corpora are scanned to collect context elements c. However, only those context elements that can be found in t are encoded into models. If t contains an identifier sym-

bol s for a given context element c, then s is passed to Algorithm 1 to update vector $\vec{w}$.

3 Reports from Empirical Evaluation

3.1 General Settings

As input, we use the Wikipedia text corpora provided by the task organizers.[2] In our reports, we include results from the sense-based NASARI vectors (i.e., the baseline introduced by the organizers): 300-dimensional embeddings obtained using a hybrid approach (Camacho-Collados et al., 2016). The evaluation metric is the harmonic mean (H) of Pearson's r and Spearman ρ correlations between the test datasets (i.e., gold data constructed from scores assigned by humans to word pairs) and the corresponding system generated ones.

We treat multi-word expressions similar to single-token words. Given a list of tokens, instead of collecting co-occurrence information only for single tokens, we extend our scan of input corpora to contiguous n-gram sequences of tokens for which n is decided by the maximum length of items in the evaluation test sets. In effect, we limit the active vocabulary of our system and collect co-occurrence information only for those lexical items in the task's test sets.

3.2 Monolingual Subtask

To collect co-occurrence information from input corpora, given the small size of input corpora, we adapt a greedy approach. Input corpora are read line by line; if a lexical item w_t in our target vocabulary appears in a line at span i to j, we update $\vec{w_t}$ by passing the following items as context element to Algorithm 1:

Feature Sets:

- The whole line (as one unit): this is done to capture information about possible co-occurrences of test lexical items within a large context (such as done in word-by-document models).
- All the tokens from position $i - 20$ to $j + 20$ (i.e., including w_t), i.e., the classic sliding context window. We include w_t to enforce similarity between a pair of multiword lexical items of similar constituent tokens.
- All n-grams ($n \in \{3, 4\}$) generated from each of the tokens appearing in the above sliding context window: this is done to capture information about the morphological structure of the context words.

[2] https://sites.google.com/site/rmyeid/projects/polyglot

Lang	r	ρ	H	m	Weighting	Similarity	RUN
FA	.541	.585	.562	2000	Cascaded-PPMI	r	1
FA	.606	.601	.604	2500	Cascaded-PPMI	sim_{lin}	2
EN	.71	.699	.704	2500	Cascaded-PPMI	sim_{lin}	1
EN	.656	.697	.676	2500	Cascaded-PPMI	r	2

Table 1: Results for our official submissions.

Table 1 summarizes the results and configurations that we have used in our official submissions. For Farsi, for the first run, we built vectors of dimension $m = 2000$, weighted them using cascaded-PPMI (see Section 2.1) and used Pearson's r as a similarity measure. Evaluated by the organisers, this resulted in $r = 0.541$, $\rho = 0.585$, and the official score of $H = 0.562$. In the second run, however, we built vectors of dimensionality $m = 2500$ and after cascaded-PPMI weighting, similarities were computed using sim_{lin}. This resulted in scores of $r = 0.606$, $\rho = 0.601$, and $H = 0.604$. To choose these configurations, we relied on the trial data as well as resources introduce in Camacho-Collados et al. (2015). For English, we observed that adding n-gram features deteriorates results; hence, we removed this set of features from our model of dimensionality $m = 2500$. In both runs, we used cascaded-PPMI. As a similarity measure, we used sim_{lin} and Pearson's r in the first and second run, respectively. This produced a score of $r = 0.71$, $\rho = 0.699$, and $H = 0.704$ for the first run, and $r = 0.656$, $\rho = 0.697$, and $H = 0.676$ over the second run. Note that for both languages, we could build any vectors for a number lexical items since they did not occur in the input corpora (see the last column of Table 2 for details).

3.2.1 Extended Evaluations

While our official submissions are limited to English and Farsi, to provide a better understanding of the method's performance, we provide results for all the five languages in the monolingual subtask. To build models, we use the feature sets described in the previous section. The remaining hyper-parameter of our method is m (the dimensionality of models); we report results for $m \in \{300, 700, 2000\}$. Results obtained using various

Lang	Baseline			Hash Method - WPPMI - SimPearson									#M
				Dim =300			Dim = 700			Dim = 2000			
	r	ρ	H	r	ρ	H	r	ρ	H	r	ρ	H	
DE	0.513	0.514	**0.514**	0.439	0.436	**0.438**	0.537	0.574	**0.555** ↑	0.603	0.655	**0.628** ↑	16
EN	0.683	0.681	**0.682**	0.428	0.474	**0.450**	0.535	0.598	**0.564**	0.614	0.652	**0.632**	3
ES	0.602	0.597	**0.600**	0.512	0.568	**0.539**	0.576	0.644	**0.608** ↑	0.665	0.719	**0.691** ↑	7
FA	0.412	0.398	**0.405**	0.475	0.496	**0.486** ↑	0.512	0.535	**0.523** ↑	0.538	0.569	**0.553** ↑	25
IT	0.597	0.594	**0.596**	0.469	0.503	**0.485**	0.537	0.589	**0.562**	0.616	0.674	**0.643** ↑	12

Table 2: Results for vectors of various dimensionality (denoted by dim), and when using PPMI for weighting and Pearson's r for measuring similarity between them. H denotes the harmonic mean of r and ρ (i.e., the task's official score). #M is the number of lexical items which have not occurred in our input corpora; for pairs containing these items, we use 0 as a default value for similarity. Those settings that yield better results than the baseline are marked using ↑.

Lang	Dim =300			Dim = 700			Dim = 2000		
	r	ρ	H	r	ρ	H	r	ρ	H
DE	.576	.577	**.576** ↑	.609	.609	**.609** ↑	.619	.617	**.618** ↑
EN	.633	.627	**.630**	.659	.653	**.656**	.644	.633	**.638**
ES	.660	.659	**.659** ↑	.675	.670	**.673** ↑	.669	.669	**.669** ↑
FA	.449	.439	**.444** ↑	.468	.458	**.463** ↑	.517	.506	**.512** ↑
IT	.609	.601	**.605** ↑	.617	.611	**.614** ↑	.618	.612	**.615** ↑

Table 3: Method's performance when using PPMI for weighting and Goodman and Kruskal's γ for a similarity measurement. This combination gives the best performance for models of small dimensionality such as $m = 300$.

Lang	Dim =300			Dim = 700			Dim = 2000		
	r	ρ	H	r	ρ	H	r	ρ	H
DE	.392	.377	**.384**	.511	.515	**.513**	.616	.624	**.620** ↑
EN	.435	.436	**.436**	.548	.553	**.551**	.632	.630	**.631**
ES	.506	.505	**.506**	.583	.578	**.580**	.673	.683	**.678** ↑
FA	.477	.501	**.488** ↑	.518	.540	**.529** ↑	.551	.573	**.562** ↑
IT	.445	.443	**.444**	.532	.534	**.533**	.643	.650	**.646** ↑

Table 4: Method's performance when using the combination of PPMI and sim_{lin}.

combinations of weighting techniques and similarity measure are summarized in Table 2 to 7.[3]

Disregarding the choice of weighting technique and similarity measure, an increase in m often produces better results, but at the expense of higher computational cost. In addition, as suggested in Section 2.1, by comparing results between Table 2 to 4 and Table 5 to 7, we observe that using cascaded-PPMI weighting instead of simple PPMI weighting often yields better scores. The

[3]Slight improvements in results for Farsi are due to homogenizing character encoding: Zero-width non-joiner characters (U+200c) are replaced by the space character (U+0020); the Arabic letter Kaf (U+0643) is replaced by the Farsi letter Kaf U+06A9, and the Arabic letter Yeh (U+064A) is replaced by the Farsi letter Yeh (U+FBFC).

Lang	Dim =300			Dim = 700			Dim = 2000		
	r	ρ	H	r	ρ	H	r	ρ	H
DE	.486	.486	**.486**	.587	.626	**.606** ↑	.630	.675	**.651** ↑
EN	.519	.538	**.528**	.608	.647	**.627**	.639	.668	**.653**
ES	.572	.626	**.598**	.646	.695	**.670** ↑	.683	.721	**.701** ↑
FA	.507	.521	**.514** ↑	.535	.565	**.550** ↑	.552	.595	**.573** ↑
IT	.516	.538	**.527**	.597	.638	**.617** ↑	.626	.670	**.647** ↑

Table 5: Method's performance when using the combination of cascaded-PPMI and Pearson's r.

Lang	Dim =300			Dim = 700			Dim = 2000		
	r	ρ	H	r	ρ	H	r	ρ	H
DE	.551	.556	**.553** ↑	.630	.633	**.631** ↑	.648	.652	**.650** ↑
EN	.608	.603	**.606**	.659	.653	**.656**	.661	.648	**.655**
ES	.647	.650	**.649** ↑	.692	.688	**.690** ↑	.688	.684	**.686** ↑
FA	.500	.487	**.493** ↑	.528	.517	**.523** ↑	.559	.551	**.555** ↑
IT	.593	.598	**.595**	.640	.633	**.636** ↑	.637	.631	**.634** ↑

Table 6: Method's performance when using the combination of cascaded-PPMI and γ.

Lang	Dim =300			Dim = 700			Dim = 2000		
	r	ρ	H	r	ρ	H	r	ρ	H
DE	.434	.454	**.444**	.597	.614	**.605** ↑	.665	.686	**.675** ↑
EN	.497	.508	**.502**	.641	.644	**.643**	.684	.677	**.680**
ES	.575	.589	**.582**	.677	.683	**.680** ↑	.727	.733	**.730** ↑
FA	.537	.557	**.547** ↑	.582	.592	**.587** ↑	.589	.605	**.597** ↑
IT	.513	.513	**.513**	.634	.641	**.637** ↑	.690	.693	**.692** ↑

Table 7: Method's performance for the combination of cascaded-PPMI and sim_{lin}: This combination proves to provide the best results for high-dimensional models.

Lang	r	ρ	H	RUN
EN-FA	0.519	0.492	0.505	Baseline
EN-FA	0.485	0.544	0.513	1
EN-FA	0.429	0.582	0.494	2

Table 8: Results for EN-FA detest.

only exception is when m is small (e.g., $m = 300$) and γ is used to measure similarities. For small $m = 300$, this combination of PPMI weighting and γ gives the best performance (Table 3); we witness that for $m = 300$, this combination also gives the best results for Camacho-Collados et al.'s data sets.

3.3 Cross-Lingual Subtask

We applied the methodology described in Section 2.2 to build cross-lingual embeddings for the pair emphEnglish and Farsi. To build the common symbol space, we extracted an English-to-Farsi translation dictionary from the English Wiktionary dump of January 2017, containing translations for 7500 lexical items in English. These 7500 entries were converted to a symbol table t of size 17760. We then augmented this table with Wikipedia's title translations. As a result, the number of entries in t increased to 1,299,770.

For each w in the test data set, we collected co-occurrences from a context window (extended 20 tokens at each side of w) for both words and multiword expressions that appear in t. Note that the sole input to our method was unaligned text from the English and Farsi Wikipedia corpus (similar to the monolingual setting). In both runs, we used vectors of dimensionality $m = 3000$ and the proposed sim_{lin} measure to compute similarities between vectors. To weight vectors, in the first run, we used cascaded-PPMI while we used simple PPMI for the second run. Table 8 provides a summary of the method's performance. Surprisingly, our simple methodology performs at least as well as the baseline technique.

Results reported in Table 8 can be easily improved by feeding in additional input, particularly parallel corpora. For instance, we observe that using the Open Subtitles corpus in addition to the Wikipedia corpus can enhance the results for the combination of cascaded-PPMI and sim_{lin} (Run 1) from $H = 0.505$ to 0.575.

4 Conclusion

This paper described the methodology behind the HHU system that participated in the SemEval 2017 shared task on semantic word similarity. The proposed technique uses a hash-based algorithm for building embeddings. The method is fast and simple, and it demands only a small amount of computational resources to build a model. As shown by empirical evaluations, our method shows acceptable performance in semantic similarity tasks. Our code is available for download (`https://user.phil.hhu.de/~zadeh/material/hash-vectors/`) in order to replicate the results reported in this paper.

Acknowledgments

This work was supported by the CRC 991 "The Structure of Representations in Language, Cognition, and Science" funded by the German Research Foundation (DFG). We would like to thank Matthias Liebeck and Pashutan Modaresi for their comments and suggestions.

References

Jose Camacho-Collados, Mohammad Taher Pilehvar, Nigel Collier, and Roberto Navigli. 2017. 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, Vancouver, Canada, pages 15–26. http://www.aclweb.org/anthology/S17-2002.

José Camacho-Collados, Mohammad Taher Pilehvar, and Roberto Navigli. 2015. A unified multilingual semantic representation of concepts. 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)*. Association for Computational Linguistics, Beijing, China, pages 741–751. http://www.aclweb.org/anthology/P15-1072.

José Camacho-Collados, Mohammad Taher Pilehvar, and Roberto Navigli. 2016. Nasari: Integrating explicit knowledge and corpus statistics for a multilingual representation of concepts and entities. *Artif. Intell.* 240:36–64. https://doi.org/10.1016/j.artint.2016.07.005.

Peter Y. Chen and Paula M. Popovich. 2002. *Correlation: Parametric and Nonparametric Measures*. Quantitative Applications in the Social Sciences. Sage Publications.

Kenneth Ward Church and Patrick Hanks. 1990. Word association norms, mutual information, and

lexicography. *Comput. Linguist.* 16(1):22–29. http://dl.acm.org/citation.cfm?id=89086.89095.

Leo A. Goodman and William H. Kruskal. 1954. Measures of association for cross classifications. *Journal of the American Statistical Association* 49(268):732–764. http://www.jstor.org/stable/2281536.

Zellig S Harris. 1954. Distributional structure. *Word* 10(2-3):146–162.

Dekang Lin. 1998. An information-theoretic definition of similarity. In *Proceedings of the Fifteenth International Conference on Machine Learning*. Morgan Kaufmann Publishers Inc., San Francisco, CA, USA, ICML '98, pages 296–304. http://dl.acm.org/citation.cfm?id=645527.657297.

B. QasemiZadeh, L. Kallmeyer, and P. Passban. 2017. Sketching Word Vectors Through Hashing. *ArXiv e-prints* abs/1002.0035. https://arxiv.org/abs/1705.04253v1.

Behrang QasemiZadeh and Laura Kallmeyer. 2016. Random positive-only projections: PPMI-Enabled incremental semantic space construction. In *Proceedings of the Fifth Joint Conference on Lexical and Computational Semantics*. Association for Computational Linguistics, Berlin, Germany, pages 189–198. http://anthology.aclweb.org/S16-2024.

Behrang QasemiZadeh, Laura Kallmeyer, and Aurelie Herbelot. 2017. Projection aleatoire non-negative pour le calcul de word embedding. In *Proceedings of TALN 2017*.

Peter D. Turney. 2001. Mining the web for synonyms: PMI-IR versus LSA on TOEFL. In *Proceedings of the 12th European Conference on Machine Learning*. Springer-Verlag, London, UK, UK, EMCL '01, pages 491–502. http://dl.acm.org/citation.cfm?id=645328.650004.