

On metric embedding for boosting semantic similarity computations

Abstract

Computing pairwise word semantic similarity is widely used and serves as a building block in many tasks in NLP. In this paper, we explore the embedding of the shortest-path metrics from a knowledge base (Wordnet) into the Hamming hypercube, in order to enhance the computation performance. We show that, although an isometric embedding is untractable, it is possible to achieve good non-isometric embeddings. We report a speedup of three orders of magnitude for the task of computing Leacock and Chodorow (LCH) similarities while keeping strong correlations ($r = .819, \rho = .826$).

1 Introduction

Among semantic relatedness measures, semantic similarity encodes the conceptual distance between two units of language – this goes beyond lexical resemblance. When words are the speech units, semantic similarity is at the very core of many NLP problems. It has proven to be essential for word sense disambiguation (Mavroudis et al., 2005; Basile et al., 2014), open domain question answering (Yih et al., 2014), and information retrieval on the Web (Varelas et al., 2005), to name a few. Two established strategies to estimate pairwise word semantic similarity includes knowledge base and distributional semantics.

Knowledge base approaches exploit the structure of the taxonomy ((Leacock and Chodorow, 1998; Hirst and St-Onge, 1998; Wu and Palmer, 1994)), its content ((Banerjee and Pedersen, 2002)), or both (Resnik, 1995; Lin, 1998). In the earliest applications, Wordnet-based semantic similarity played a predominant role so that semantic similarity measures reckon with on the information from the lexical hierarchy. It there-

fore ignores contextual information on word occurrences and relies on humans to encode such hierarchies – a tedious task in practice. In contrast, well-known distributional semantics strategy encodes semantic similarity using the correlation of statistical observations on the occurrences of words in a textual corpora (Lin, 1998).

While providing a significant impact on a broad range of applications (Herbelot and Ganesalingam, 2013; Lazaridou et al., 2013; Beltagy et al., 2014; Bernardi et al., 2013; Goyal et al., 2013), distributional semantics similar to knowledge base strategy – struggle to process the ever-increasing size of textual corpora in a reasonable amount of time. As an answer, embedding high-dimensional distributional semantics models for words into low-dimensional spaces (henceforth *word embedding* (Collobert and Weston, 2008)) has emerged as a popular method. Word embedding utilizes deep learning to learn a real-valued vector representation of words so that any vector distance – usually the cosine similarity – encodes the word-to-word semantic similarity. Although word embedding was successfully applied for several NLP tasks (Hermann et al., 2014; Andreas and Klein, 2014; Clinchant and Perronnin, 2013; Xu et al., 2014; Li and Liu, 2014; Goyal et al., 2013), it implies a slow training phase – measured in days (Collobert and Weston, 2008), though re-embedding words seems promising (Labutov and Lipson, 2013). There is another usually under-considered issue: the tractability of the pairwise similarity computation in the vector space for large volume of data. It is linear in time with a limited data locality advantage since the two vectors are traversed in parallel. Despite these limitations, the current enthusiasm for word embedding certainly echoes the need for lightning fast word-to-word semantic similarity computation.

In this context, it is surprising that embedding semantic similarity of words in low dimensional

spaces for knowledge base approaches is understudied. This oversight may well condemn the word-to-word semantic similarity task to remain corpus-dependant – i.e. ignoring the background knowledge provided by a lexical hierarchy.

In this paper, we propose an embedding of knowledge base semantic similarity based on the shortest path metric (Leacock and Chodorow, 1998), into the Hamming hypercube of size n (the size of targeted binary codes).

This technique benefits from the execution of hamming distance within a few cycles on modern CPUs. This allows the computation of several millions distances per second. However, the dimension of the hypercube (i.e. the number of bits used to represent an element) should obey the threshold of few CPU words (64, 128 . . . , bits) to maintain such efficiency (Heo et al., 2012) to maintain such efficiency.

An isometric embedding requires excessively higher number of dimensions to deemed to be feasible. However, in this paper we show that practical embeddings exist and present a method to construct them. The best embedding presents very strong correlations ($r = .819, \rho = .829$) with the Leacock & Chodorow similarity measure (LCH in the rest of this paper). Meanwhile performance is increased by up to three orders of magnitude.

2 Shortest path metric embedding

Let us first introduce few notations. We denote H_2^n as an n -dimensional hypercube whose nodes are labeled by the 2^n binary n -tuples. The nodes are adjacent if and only if their corresponding n -tuples differ in exactly one position, i.e. their Hamming distance (ℓ_1) is equal to one. In what follows, Q^n denotes the metric space composed of H_2^n with ℓ_1 .

We tackle the following problem: We aim at defining a function f that maps every node w of the taxonomy (Wordnet for Leacock and Chow) into Q^n so that for every pair of nodes: $\forall(w_i, w_j), d(w_i, w_j) = \lambda \cdot \ell_1(f(w_i), f(w_j))$, where λ is a scalar. For practical usage, the construction of the mapping should also be reasonable in terms of time complexity.

Theoretical limitations Wordnet with its hypernym relation forms a partially ordered set (poset). The first approach is to perform an isometric embedding from the poset with shortest path distance into the Hamming hypercube. Such a mapping would exactly preserve the original distance in the

embedding. As proven by (Deza and Laurent, 1997), poset lattices, with their shortest path metric, can be isometrically embedded into the hypercube, but the embedding requires 2^n dimensions. The resulting embedding would not fit in the memory of any existing computer, for a lattice having more than 60 nodes, . Using Wordnet, with tens of thousands synsets, this embedding is untractable. The bound given by Deza et al. is not tight, however it would require a more than severe improvement to be of any practical interest.

Tree embedding To reduce the dimensionality, we weaken the lattice into a tree. We build a tree from the Wordnet’s Hyponyms/Hypernyms poset by cutting 1,300 links, which corresponds to roughly one percent of the edges in the original lattice. The nature of the cut to be performed can be subject to discussion. In this preliminary research, we used a simple approach. Since hypernyms are ordered, we decided to preserve only the first hypernym – semantically more relevant, or at least statistically – and to cut edges to other hypernyms. Our experiments in Table 1 shows that using the obtained tree instead of the lattice keeps a high correlation ($r = .919, \rho = .931$) with the original LCH distance, thus validating the approach.

(Wilkeit, 1990) showed that any k -ary tree can be embedded into Q^{n-1} . We give an isometric embedding algorithm linear in time and space, faster than Winkler’s generic approach for graphs in $O(n^5)$ (Winkler, 1984). The algorithm is the following: at each step of a depth-first pre-order traversal: if the node has k children, we set the signature for the i -th child by appending k zeroes to the parent’s signature and by setting the i -th of the k bits to one. An example is given in Figure 1. The dimension reduction to tens of kilobits per node remains far from our goal of several CPU words.

Looking at the construction of the isometric em-

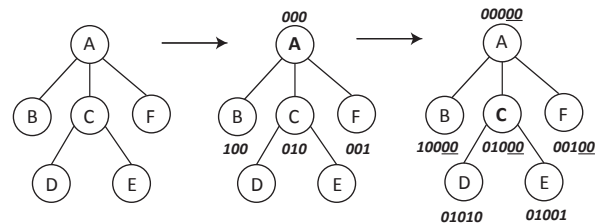


Figure 1: Construction of isometric embedding on a sample tree. For this six nodes tree, the embedding requires five bits.

bedding, the large dimension results from the appending of bits to all nodes in the tree. This results in a large number of bits that are rarely set to one. At the opposite, the optimal embedding in terms of dimension is given by the approach of (Chen and Stallmann, 1995) that assigns gray codes to each node. However, the embedding is not isometric and introduces a very large error. As shown in Table 1, this approach gives the most compact embedding with $\lceil \log_2(87,000) \rceil = 17$ bits, but leads to poor correlations ($r = .235$ and $\rho = .186$).

An exhaustive search is also out of reach: for a fixed dimension n and r nodes in the tree, the number of combinations C is given by:

$$C = \frac{(2^n)!}{(n-r)!}$$

Even with the smallest value of $n = 17$ and $r = 87,000$, we have $C > 10^{10,000}$. With $n = 64$, to align to a CPU word, $C > 10^{100,000}$.

3 Non-isometric Embedding

Our approach is a trade-off between the isometric embedding and the pre-order gray code solution. When designing our algorithm, we had to decide which tree distance we will preserve, either between parent and children, or among siblings.

Therefore, we take into account the nature of the tree we utilized for the embedding we devise. Let first analyse the characteristics of the tree obtained from the cut. The tree has an average branching factor of 4.9, with a standard deviation of 14 and 96% of the nodes have a branching factor lesser than 20. At the opposite, the depth is very stable with an average of 8.5, a standard deviation of 2, and a maximum of 18. Consequently, we decide to preserve the parent-children distance over the very unstable siblings distance. To lower the dimensions, we aim at allocating less than k bits for a node with k children, and at avoiding the signature extension taking place for every node in the isometric approach. Our approach uses the following principles.

Branch inheritance: each node inherits the signature from its father, but contrary to isometric embedding, the signature extension does not apply to all the nodes in the tree. This guarantees the compactness of the structure.

Parentship preservation: when allocating less bits than required for the isometric embedding, we introduce an error. Our allocation favours as much

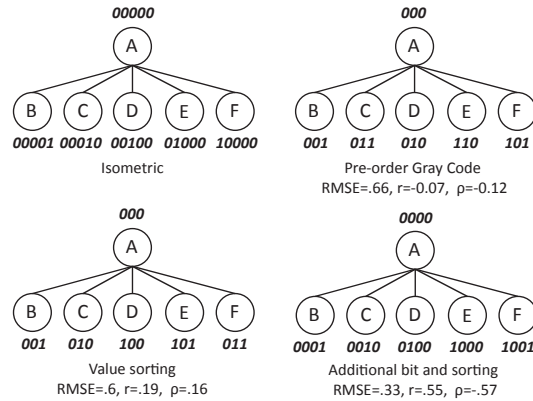


Figure 2: Approaches to reduce the tree embedding dimensions.

as possible the parentship distance at the expense of the sibling distance. As a first allocation, for a node with k children, we allocate $\lceil \log_2(k+1) \rceil$ bits for the signatures, in order to guarantee the unicity of the signature.

Word alignment: The dimension D of the embedding is not necessarily aligned to a CPU word size W : $kW \leq D \leq (k+1)W$. We want to exploit the potential $(k+1)W - D$ bits that are unused but still processed by the CPU. For this purpose we rank the nodes along a value $v(i), i \in N$ to decide which nodes are allowed to use extra bits. Since our approach favours parent/child distance, we want to allow additional bits for nodes that are both close to the root and the head of a large branch. To balance the two values, we use the following formula: $v(i) = (max_{depth} - depth(i)) \cdot \log(size_{branch}(i))$

We therefore enable our approach to take full advantage of the otherwise unused bits.

To enhance the quality of the embedding, we also introduce two potential optimisations. The first is *Children-sorting*: we allocate a better preserving signature to children having larger descents. The second is *Value-sorting* and is depicted in Figure 2. Among the $2^{\lceil \log_2(k+1) \rceil}$ available signatures, only $k+1$ will be used. For instance in the case of 5 children as depicted in Figure 2, we allocate 3 bits for 6 signatures. We favor the parentship distance by selecting first the signatures where one bit differs from the parent's one.

4 Experiments

We run two experiments to evaluate both the soundness and the performance of our approach. In the first experiment, we test the quality of our embedding against the tree distance and the LCH

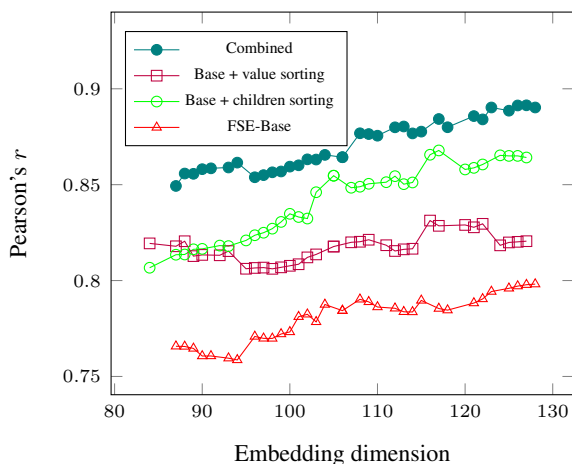


Figure 3: FSE: influence of optimizations and dimensions on the correlation over the tree distance on Wordnet.

similarity. In the second experiment we measure the speedup over standard libraries.

Our algorithm called FSE for Fast Similarity Embedding, is implemented in Java and available publicly¹. Our testbed is an Intel Xeon E3 1246v3 with 16GB of memory.

4.1 Embedding

First we measure the correlation of the embedded distance with the original tree distance, to assess the value of the optimizations. Figure 3 shows the influence of dimensions and optimisations on the Pearson's product moment correlation r . The base version reaches $r = .77$, combined optimizations exhibit a higher correlation ($r = .89$).

We then measure the correlation with the Leacock & Chodorow similarity measure. We compute the correlation on 5 millions distances from the Wordnet-Core noun pairs² (Table 1). Similarly to the results obtained on the tree distance correlation, FSE-Best exhibits the highest scores with $r = .819$ and $\rho = .829$, not far from the bound of $r = .919$ and $\rho = .931$ for the isometric embedding of the same tree. Our approach requires 650 times less bits than the isometric one, while keeping strong guarantees on the correlation with the original LCH distance.

¹Source code, binaries and instructions to reproduce the experiments are available at <http://bit.ly/FSELCH>

²<https://wordnet.princeton.edu/wordnet/download/standoff/>

Embedding	Bits	Pearson's r	Spearman's ρ
Chen et al.	17	.235	.186
FSE-Base	84	.699	.707
FSE-Best	128	.819	.829
Isometric	84K	.919	.931

Table 1: Correlations between LCH, isometric embedding, and FSE for all distances on all Wordnet-Core noun pairs (p -values $\leq 10^{-14}$).

Algo	Measure	Amount of pairs (n)				
		10^3	10^4	10^5	10^6	10^7
WS4J	10^3 . ms	0.156	1.196	11.32	123.89	1,129.3
FSE-Best	ms	0.04	0.59	14.15	150.58	1,482
speedup		$\times 3900$	$\times 2027$	$\times 800$	$\times 822$	$\times 762$

Table 2: Running time in milliseconds for pairwise similarity computations.

4.2 Speedup

Table 4.2 presents the computation time of the LCH similarity. This is computed using WS4J³, an efficient library that enables in-memory caching.

Because of the respective computational complexities of the Hamming distance and the shortest path algorithms, FSE unsurprisingly boosts LCH similarity computation by orders of magnitudes. When the similarity is computed on a small number of pairs (a situation of the utmost practical interest), the factor of improvement is three orders of magnitude. This factor increases to an amount of 800 times for an application in a steady state.

5 Conclusion

We proposed in this paper a novel approach based on metric embedding to boost the computation of shortest-path based similarity measures such as the one of Leacock & Chow. We showed that an isometric embedding of the Wordnet's hypernym/hyponym lattice does not lead to a practical solution. To tackle this issue, we weaken the lattice structure into a tree by cutting less relevant edges. We then devised an algorithm and several optimizations to embed the tree shortest-path distance in a word-aligned number of bits. Such an embedding can be used to boost NLP core algorithms – this was demonstrated here on the computation of LCH for which our approach offers a factor of improvement of three orders of magnitude, with a very strong correlation.

³<https://code.google.com/p/ws4j/>

References

- Jacob Andreas and Dan Klein. 2014. How much do word embeddings encode about syntax. In *Association for Computational Linguistics (ACL)*.
- Satanjeev Banerjee and Ted Pedersen. 2002. An adapted lesk algorithm for word sense disambiguation using wordnet. In *Computational linguistics and intelligent text processing*, pages 136–145. Springer.
- Pierpaolo Basile, Annalina Caputo, and Giovanni Semeraro. 2014. An enhanced lesk word sense disambiguation algorithm through a distributional semantic model. In *Proceedings of COLING*, pages 1591–1600.
- Islam Beltagy, Katrin Erk, and Raymond Mooney. 2014. Semantic parsing using distributional semantics and probabilistic logic. *Association for Computational Linguistics (ACL)*, page 7.
- Raffaella Bernardi, Georgiana Dinu, Marco Marelli, and Marco Baroni. 2013. A relatedness benchmark to test the role of determiners in compositional distributional semantics. In *Association for Computational Linguistics (ACL)*, pages 53–57.
- Woei-Kae Chen and Matthias FM Stallmann. 1995. On embedding binary trees into hypercubes. *Journal of Parallel and Distributed Computing*, 24(2):132–138.
- Stéphane Clinchant and Florent Perronnin. 2013. Aggregating continuous word embeddings for information retrieval. *Association for Computational Linguistics (ACL)*.
- Ronan Collobert and Jason Weston. 2008. A unified architecture for natural language processing: Deep neural networks with multitask learning. In *Proceedings of the 25th international conference on Machine learning*, pages 160–167. ACM.
- M. Deza and M. Laurent. 1997. *Geometry of Cuts and Metrics*. Springer, 588 pages.
- Kartik Goyal, Sujay Kumar Jauhar, Huiying Li, Mrinmaya Sachan, Shashank Srivastava, and Eduard H Hovy. 2013. A structured distributional semantic model for event co-reference. In *Association for Computational Linguistics (ACL)*, pages 467–473.
- Jae-Pil Heo, Youngwoon Lee, Junfeng He, Shih-Fu Chang, and Sung-Eui Yoon. 2012. Spherical hashing. In *Computer Vision and Pattern Recognition (CVPR), 2012 IEEE Conference on*, pages 2957–2964. IEEE.
- Aurélie Herbelot and Mohan Ganesalingam. 2013. Measuring semantic content in distributional vectors. In *Association for Computational Linguistics (ACL)*, pages 440–445.
- Karl Moritz Hermann, Dipanjan Das, Jason Weston, and Kuzman Ganchev. 2014. Semantic frame identification with distributed word representations. In *Association for Computational Linguistics (ACL)*.
- Graeme Hirst and David St-Onge. 1998. Lexical chains as representations of context for the detection and correction of malapropisms. *WordNet: An electronic lexical database*, 305:305–332.
- Igor Labutov and Hod Lipson. 2013. Re-embedding words. In *ACL (2)*, pages 489–493.
- Angeliki Lazaridou, Marco Marelli, Roberto Zamparelli, and Marco Baroni. 2013. Compositionally derived representations of morphologically complex words in distributional semantics. In *Association for Computational Linguistics (ACL)*, pages 1517–1526.
- Claudia Leacock and Martin Chodorow. 1998. Combining local context and wordnet similarity for word sense identification. *WordNet: An electronic lexical database*, 49(2):265–283.
- Chen Li and Yang Liu. 2014. Improving text normalization via unsupervised model and discriminative reranking. *Association for Computational Linguistics (ACL)*, page 86.
- Dekang Lin. 1998. An information-theoretic definition of similarity. In *ICML*, volume 98, pages 296–304.
- Dimitrios Mavroudis, George Tsatsaronis, Michalis Vazirgiannis, Martin Theobald, and Gerhard Weikum. 2005. Word sense disambiguation for exploiting hierarchical thesauri in text classification. In *Knowledge Discovery in Databases: PKDD 2005*, pages 181–192. Springer.
- Philip Resnik. 1995. Using information content to evaluate semantic similarity in a taxonomy. *arXiv preprint cmp-lg/9511007*.
- Giannis Varelas, Epimenidis Voutsakis, Paraskevi Raftopoulou, Euripides GM Petrakis, and Evangelos E Milios. 2005. Semantic similarity methods in wordnet and their application to information retrieval on the web. In *Proceedings of the 7th annual ACM international workshop on Web information and data management*, pages 10–16. ACM.
- Elke Wilkeit. 1990. Isometric embeddings in hamming graphs. *Journal of Combinatorial Theory, Series B*, 50(2):179–197.
- Peter M Winkler. 1984. Isometric embedding in products of complete graphs. *Discrete Applied Mathematics*, 7(2):221–225.
- Zhibiao Wu and Martha Palmer. 1994. Verbs semantics and lexical selection. In *Proceedings of the 32nd annual meeting on Association for Computational Linguistics*, pages 133–138. Association for Computational Linguistics.

Liheng Xu, Kang Liu, Siwei Lai, and Jun Zhao. 2014. Product feature mining: Semantic clues versus syntactic constituents. In *Association for Computational Linguistics (ACL)*, pages 336–346.

Wen-tau Yih, Xiaodong He, and Christopher Meek. 2014. Semantic parsing for single-relation question answering. In *Proceedings of ACL*.