Available online at www.sciencedirect.com

ScienceDirect

Computer Speech & Language 56 (2019) 107–129

www.elsevier.com/locate/csl

Unsupervised sentence representations as word information series: Revisiting TF–IDF

Ignacio Arroyo-Fernández*[,a], Carlos-Francisco Méndez-Cruz[1,b], Gerardo Sierra[c,2,c], Juan-Manuel Torres-Moreno[3,d], Grigori Sidorov[4,e]

[a] *Universidad Nacional Autónoma de México (UNAM), Mexico*
[b] *Centro de ciencias genómicas (CCG–UNAM), Mexico*
[c] *Instituto de ingeniería (IIngen–UNAM), Mexico*
[d] *Laboratoire Informatique d'Avignon (LIA–UAPV), Mexico*
[e] *Centro de investigación en computación (CIC–IPN), Mexico*

Received 18 October 2017; received in revised form 11 November 2018; accepted 25 January 2019
Available online 30 January 2019

Abstract

Sentence representation at the semantic level is a challenging task for natural language processing and Artificial Intelligence. Despite the advances in word embeddings (i.e. word vector representations), capturing sentence meaning is an open question due to complexities of semantic interactions among words. In this paper, we present an embedding method, which is aimed at learning unsupervised sentence representations from unlabeled text. We propose an unsupervised method that models a sentence as a weighted series of word embeddings. The weights of the series are fitted by using Shannon's Mutual Information (MI) among words, sentences and the corpus. In fact, the Term Frequency–Inverse Document Frequency transform (TF–IDF) is a reliable estimate of such MI. Our method offers advantages over existing ones: identifiable modules, short-term training, online inference of (unseen) sentence representations, as well as independence from domain, external knowledge and linguistic annotation resources. Results showed that our model, despite its concreteness and low computational cost, was competitive with the state of the art in well-known Semantic Textual Similarity (STS) tasks.

Keywords: Sentence representation; Sentence embedding; Word embedding; Information entropy; TF–IDF; Natural language processing

* Corresponding author. Present address: Ciudad universitaria, AV. Universidad No. 3000, Coyoacán 04510, Ciudad de México, Mexico.
E-mail addresses: iaf@ciencias.unam.mx (I. Arroyo-Fernández), cmendezc@ccg.unam.mx (C.-F. Méndez-Cruz), gsierram@ii.unam.mx (G. Sierra), juan-manuel.torres@univ-avignon.fr (J.-M. Torres-Moreno), sidorov@cic.ipn.mx (G. Sidorov).
[1] Present address: Av. Universidad s/n Col. Chamilpa 62210, Cuernavaca, Morelos, Mexico.
[2] Present address: Ciudad universitaria, AV. Universidad No. 3000, Coyoacán 04510, Ciudad de México, Mexico.
[3] Present address: Université d'Avignon et des Pays de Vaucluse., 339 chemin des Meinajaries 84911, Avignon cedex 9, France.
[4] Present address: Instituto Politécnico Nacional. Av. Juan de Dios Bátiz, Esq. Miguel Othón de Mendizábal, Col. Nueva Industrial Vallejo, Gustavo A. Madero 07738, Ciudad de México, Mexico.

https://doi.org/10.1016/j.csl.2019.01.005
0885-2308/

1. Introduction

Nowadays, the growth of information in digital media encourages the analysis of large amounts of text data. This is attracting attention from Data Science and Artificial Intelligence researchers, as well as from the Internet industry. Internet users are responsible for a meaningful part of this growth. They enter information into the network which is also leveraged for sharing knowledge. An important part of this knowledge is found at repositories such as question & answer forums, digital newspapers and digital encyclopedias.

Due to the innumerable duplication of the information at these repositories, several concerns arise as to the way users feed and consume knowledge. Some of these concerns include removing redundancies in question-answering forums or exploiting redundancies to assess the confidence of news in media or simply to compress text size. The accomplishment of this massive information processing is clearly infeasible for human reviewers. In this scenario, Statistical Natural Language Processing (NLP) methods are a substantial aid.

An approach to address these issues is to perform massive comparisons by considering the semantic content of sentences or short snippets of text. These comparisons can be done by measuring Semantic Textual Similarity (STS) (Hatzivassiloglou et al., 1999; Agirre et al., 2012). STS consists in computing a similarity score (a real value) between a pair of sentences. This score indicates how similar the contents of the sentences of the pair are.

There are, on the one hand, STS systems which employ a number of supervisory signals such as knowledge bases, encyclopedias, linguistic annotation resources (e.g. thesaurus and linguistic taggers built on the basis of a Part-of-Speech [PoS] tagger) and even similarity labels (Mihalcea et al., 2006). Nevertheless, for specialized texts (or for low-resourced languages) those resources are not available or are scarce. Furthermore, the scope of such systems is limited exclusively to the task of measuring textual similarity. Such a limitation generally obviates the step of representing sentence meaning. This is not desirable when we want to study statistical behavior of meaning.

On the other hand, sentences (or short text snippets) can be embedded onto vector spaces such that approximations to their meanings can be represented geometrically as vectors, i.e. sentence embeddings or sentence representations (Hinton et al., 1986; Elman, 1991). In this paper we study sentence representations because they allow for studying the statistical behavior of sentence meaning. As an additional and important benefit, sentence representations make it possible to leverage a number of NLP tasks, such as sentence clustering, text summarization (Zhang et al., 2012; Arroyo-Fernández, 2015; Arroyo-Fernndez et al., 2016; Yu et al., 2017), sentence classification (Kalchbrenner et al., 2014; Chen et al., 2017; Er et al., 2016), paraphrase identification (Yin and Schütze, 2015), semantic similarity/relatedness (Yazdani and Popescu-Belis, 2013; De Boom et al., 2016; Arroyo-Fernández and Meza Ruiz, 2017) and sentiment classification (Kalchbrenner et al., 2014; Chen et al., 2017; Onan et al., 2017).

The difficulty of representing text by embedding it into vector spaces mainly depends on its length. On the one hand, most embedding methods provide well-suited representations of the content of texts which individual size is on a relatively large scale (Salton et al., 1983; Salton and Buckley, 1988; Martin and Berry, 2007; Le and Mikolov, 2014). For instance, book-sized texts (e.g. documents with hundreds of thousands of words or larger) are well represented by the importance of the words they contain (Spärk Jones, 1972). Accordingly, by representing large text objects (e.g. as a Bag-of-Words, BoW) we can satisfy shallow (or low-complexity) information necessities limited to the gist (topics) of the documents (Manning et al., 2009; Kintsch and Mangalath, 2011), e.g. Information Retrieval and document classification.

On the other hand, words are at the bottom end of the text size scale. At the word level, information necessities can be very general. That is, word representation methods could be components of practically any NLP system. These methods are mainly based on a general principle called *distributional hypothesis*, which states that similar words are used in similar contexts (Firth, 1957; Harris, 1968). In the NLP area, this linguistic principle is usually implemented as statistical estimates of word co-occurrence, i.e. word embedding methods. These statistical estimates provide word embeddings performing well enough in general purpose NLP applications (Baroni and Lenci, 2010; Mikolov et al., 2013a; Pennington et al., 2014; Baroni et al., 2014; Bojanowski et al., 2016).

The problem of modeling sentence meaning is still open. For the cases of documents or words, most applications expect representations encoding text content or word use. Nonetheless, in the case of sentences, application users can expect composite representations providing much more specific or complex information, e.g. what is declared or denied about something (Pereira, 2000; Meza-Ruiz and Riedel, 2009; Collobert et al., 2011; Kintsch and Mangalath, 2011). Thus, state-of-the-art sentence representation methods can be highly dependent on the application and on its specificity. So it is difficult to keep their performance and behavior uniform/stable in several scenarios (Pham et al.,

2015; Pagliardini et al., 2017). Advancing the state of the art on sentence representation can be specially useful when only unlabelled text is available for learning sentence representations or even for applications to low-resourced languages.

In this work we address the problem of sentence representation by means of the following hypothesis. It is possible to represent sentences well by exploiting the link between the contexts learned by word embeddings and the entropy of such embedded words given the text containing them. In order to confirm our hypothesis, we model a sentence as a weighted series of pretrained word embeddings. In this framework, the weight of each word embedding is learned in an unsupervised fashion by measuring the Shannon's Mutual Information among its associated word, the sentence and the corpus containing it (Shannon, 1949). In this way, the weights regulate the amount of information provided by each word embedding to its corresponding sentence representation. We called this approach *Word Information Series for Sentence Embedding* (*WISSE*[5]). A good estimate of the mentioned measure is given by the well-known TF−IDF transform (Term Frequency−Inverse Document Frequency), and there are efficient implementations of it. Notice that the role the weights play in our sentence representation model is the same than that played by the attention vector of a neural language model (Er et al., 2016; Yin et al., 2016).

We evaluated our model in well-known STS tasks provided by multiple Semantic Evaluation workshops (SemEval), i.e. SICK (2014), SemEval (2016) and STS Benchmark (2017). Our results showed that WISSE performed comparably with (or outperformed) strong state-of-the-art methods in such tasks. Additional advantages were observed, which were mainly due to the modularity and low computational cost of our model: short-term training, independence from domain, external knowledge and linguistic annotation resources, as well as online inference of (unseen) sentence representations.

The rest of this paper is organized as follows: Section 2 presents the related work. Section 3 presents the main differences between STS systems and sentence representation methods. Section 4 exposes the motivations for our method. Section 5 presents the modules composing our model. In Section 6 we explain the constitution of our model and how the modules composing it interact. In Section 7 we explain the design of our experiments and their objectives. Section 8 presents the obtained results and Section 9 addresses the discussion about such results. Section 10 provides insights on possible improvements, including advantages and disadvantages. Finally, in Section 11 the conclusions derived from this work are presented.

2. Related work

In this section we mention unsupervised sentence representation models embedding short text snippets or sentences into vector spaces. Such an embedding is performed directly from unlabeled (plain) text data. The state-of-the-art methods are mainly based on neural networks. It is important to say that there exist supervised and semisupervised methods (Wieting et al., 2015; Wieting and Gimpel, 2017). Some of them are similar to our method (Arora et al., 2017; De Boom et al., 2016; Ferrero et al., 2017); however, supervised and semisupervised methods are not within the scope of this work.

2.1. Unsupervised statistical methods

A popular statistical representation method was originally used in Information Retrieval applications, by which documents were represented. The representation consists of the TF−IDF transform of document vectors such that their components are, mainly, word frequencies. Currently there are multiple heuristics for computing the components, e.g. word presence/absence (binary), smoothed logarithm, etc. (Salton et al., 1983; Salton and Buckley, 1988). The transform gives a *term-document* matrix as a result. Due to its monotonic nature, this model assigns relatively high weights to rare (infrequent) words and relatively low weights to very frequent words (Spärk Jones, 1972). According to information theory, this schema weighs the information conveyed by each word of the vocabulary (Aizawa, 2003; Robertson, 2004). The weighting method consists of a logarithmic rescaling of the frequency of each word within a document. At the end, this logarithm linearizes the exponential distribution of word types through the corpus. In our experiments, we included BoW for sentence representation as a baseline method.

[5] Our method's implementation is available at https://github.com/iarroyof/sentence_embedding.

Latent Semantic Analysis (LSA) takes as its input a term-document matrix created by means of the BoW method (Landauer et al., 1998). The sparse vectors of this matrix are transformed into document vectors, which are the projection weights to a user-defined number of eigenvectors of the term-document matrix. The transformation is computed by means of the Singular Value Decomposition (SVD) method. The number of eigenvectors is associated with the number of topics supposed to be present in the collection of documents (Martin and Berry, 2007).

2.2. Unsupervised neural network-based methods

The method called *Doc2Vec* uses a neural network to build sentence/paragraph vectors that can be used for general purposes (Le and Mikolov, 2014). This method uses word embeddings previously learned from fixed-length segments of text (sliding word windows). The words of a sentence are associated with the corresponding word embeddings (Mikolov et al., 2013b). These embeddings then are used as evidence to predict a virtual word embedding which does not represent a word, but a sentence instead.

As an extension of Doc2Vec, the neural model proposed by Kiros et al. (2015) produces sentence embeddings from the hidden states of a Recurrent Neural Network (RNN). In this framework (the Skip-thought vectors), the two sentences surrounding a center sentence are a context window. The RNN maps these sentence contexts to its last hidden state, which is taken as a sentence embedding.

The architecture called *FastSent*, proposed by Hill et al. (2016), is based on the Glove model (Pennington et al., 2014) (see Section 5.1). Furthermore, the authors instantiate the architecture of Skip-Thought vectors (Kiros et al., 2015). This combined network uses a precomputed matrix that merges co-occurrence information from both words and sentences. Additionally, Hill et al. (2016) addresses the STS problem as one of machine translation. This helps to learn sentence representations by simulating negative examples to a Sequential (Denoising) Auto Encoder, S(D) AE, which uses such examples to learn a negative model. Thus, unseen sentences can be built in opposition to the jointly learned negative (adversarial) model.

The work of Wieting et al. (2016) proposes a meaningful difference with respect to most word embedding methods. This model, called *CHARAGRAM* learns embeddings of character *n*-grams (Bojanowski et al., 2016). Character embeddings are simply averaged in order to compose words. Actually the same operation is performed when sentence embeddings are needed, i.e. the obtained word embeddings are averaged to obtain sentence representations in the *CHARAGRAM-PHRASE* model.

The neural sentence representation model called C-PHRASE relies on dependency/constituency parsing (Bentivogli et al., 2016). The idea is very similar to the one proposed by Levy and Goldberg (2014) for word embeddings (Section 5.1.2). That is, the word co-occurrence is constrained by structure dependencies rather than by word-context windows scanning the input corpus.

The model called *Sent2Vec* is similar to that proposed by Le and Mikolov (2014). The authors extended Doc2Vec for considering sentences, instead of fixed-length context windows. Additionally, this model can consider word *n*-grams or even the dynamic length of the context window as a modification of the subsampling approach proposed by Mikolov et al. (2013a). This model is also very similar to that proposed by Bojanowski et al. (2016) for word embeddings (Section 5.1.3). That is, the architecture can learn a distribution of labels for a given training example.

Although sentence representation methods based purely on deep learning have shown competitive performance based on a number of benchmarks, their computational cost can be a significant bottleneck. Some of these methods need large amounts of data and even weeks of training on GPUs to perform reasonably well (Kiros et al., 2015).

Most state-of-the-art methods are purely neural network-based. Unlike such an approach, our method uses neural models only for training word embeddings (our word embedding module). In this sense, the work of Arora et al. (2017) is very similar to ours. The authors use weighted sums of neural word embeddings to represent sentences. The weights were learned as part of a multinomial distribution. This multinomial is parameterized according to the probability of a word appearing along with other words (i.e. the words of a sentence). This idea is actually an extrapolation of the principles behind distributed representations for word embeddings at the sentence level (Mikolov et al., 2013b). In addition, this model considers a balance (a linear convex combination) between the probability of a word to occur within a *discourse* and its probability to occur within a sentence (which we find methodologically similar to TF–IDF).

3. STS and the distinction between STS systems and sentence representation methods

Since we evaluated our sentence representation method by means of STS tasks, we briefly explore such tasks' context in this section. Furthermore, we describe the relationship between STS tasks and sentence representation methods (like the proposed one in this paper) and between STS tasks and STS systems. This helps us to avoid confusions between the purposes of STS systems and sentence representation methods although they can be evaluated similarly.

STS tasks. In STS tasks, the objective is to assess the correlation score between the similarity computed by some STS algorithm and the similarity annotated by humans on a given dataset. The higher the score is, the higher the measured semantic similarity (Agirre et al., 2012). This score (the Pearson's coefficient) is a real number $\rho(\cdot\,,\cdot) \in [-1, 1]$. It is used in the task as follows. Let $\widehat{Y} = \{d(S_a^{(1)}, S_b^{(1)}), \ldots, d(S_a^{(\ell)}, S_b^{(\ell)})\}$ be the similarities of a dataset of ℓ pairs of sentences computed by some STS algorithm $d(\cdot\,,\cdot)$. Also let $Y = \{y_1, \ldots, y_\ell\}$ be the gold standard of similarities manually annotated for the pairs $(S_a^{(1)}, S_b^{(1)}), \ldots, (S_a^{(\ell)}, S_b^{(\ell)})$. The algorithm $d(\cdot\,,\cdot)$ can be evaluated by computing $\rho(\widehat{Y}, Y)$.

STS system. An STS system takes directly sentence pairs $(S_a^{(1)}, S_b^{(1)}), \ldots, (S_a^{(\ell)}, S_b^{(\ell)}) \in D$, then it learns or computes a relationship between the items of the pairs. As $\rho(\widehat{Y}, Y)$ approaches 1.0, it means that the STS algorithm computing $\widehat{Y}$ learned or computed a good relationship between the items of the pairs in the dataset D, and for the task at hand. Conversely, as $\rho(\widehat{Y}, Y)$ approaches 0.0 it means the algorithm cannot relate sentence meaning for the task at hand. Thus, the core problem of the STS system focuses on designing the such an algorithm.

Evaluating sentence embeddings via STS tasks. When evaluating sentence representations, the STS algorithm reduces to a similarity or distance measure defined on a vector space where the representations live in. For instance, the cosine similarity: $d(s_a, s_b) = \langle s_a, s_b \rangle / \| s_a \| \cdot \| s_b \|$. In this scenario where the STS algorithm is quite simple, the core problem moves to the details involved in the construction of the representations. From a mathematical view, this problem consists in having sentence vectors representing consistently sentence meaning with respect to the vector space the similarity is defined on. Consistency means, qualitatively, that dissimilar vectors must represent dissimilar sentence meanings and vice versa. Thus, a set of similarities $\widehat{Y} = \{d(s_a^{(1)}, s_b^{(1)}), \ldots, d(s_a^{(\ell)}, s_b^{(\ell)})\}$ is obtained from comparing a set of ℓ pairs of sentence representations. Herein, we have that the sentence embedding method $f(S_a) \mapsto s_a$ produces sentence representations like s_a from the input sentence S_a. Thus, we evaluate the representations by using an STS task posed by the dataset D. Therefore, as $\rho(\widehat{Y}, Y)$ approaches 1.0, it means that our sentence representation method $f(\cdot)$ embedding the representations $s_{(\cdot)} \in \mathbb{R}^d$ performs consistently on the task, with respect to sentence meaning, with respect to the vector space where $s_{(\cdot)}$ lives, and with respect to the cosine similarity algorithm $d(\cdot\,,\cdot)$.

4. On the intuitions behind weighted series of word embeddings

Currently there are multiple approaches for building sentence representations in vector spaces. Nonetheless, the complexity of the problem of keeping stable sentence representations through multiple scenarios remains a bottleneck in the NLP area. This is mainly because there exist linguistic and knowledge resources unique for a limited subset of information necessities and languages. So research on efficient unsupervised sentence representation methods can offer a promising approach to such a limitation. In this section we present theoretical intuitions that motivated our contribution.

4.1. Composition in distributional semantics

Mitchell and Lapata (2010) propose a number of candidate models for semantic composition that are empirically tested as heuristics yielding promising results in semantic representation of phrases. Among the candidate models, the asymmetric composition is particularly interesting for us. This model is a weighted sum of word embeddings. Its purpose is to approximate the meaning compositionality of short phrases (e.g. "random variable", "meaning

composition"). In this framework, the asymmetry is posed as a linguistic feature such that the *head*[H] of a phrase must be more important than the *dependent modifier*[DM]. See examples 1 and 2:

1. *random*[DM] *variable*[H]
2. *meaning*[DM] *composition*[H].

The asymmetric composition model is given by:

$$p = \alpha x_{[\mathrm{DM}]} + \beta x_{[\mathrm{H}]}, \tag{1}$$

where it must be verified that $\alpha < \beta$. This inequality reflects the difference between the importance of the constituents (i.e. the word embeddings $x_{[\mathrm{DM}]}, x_{[\mathrm{H}]} \in \mathbb{R}^n$) of the resulting phrase $p \in \mathbb{R}^n$. According to Tian et al. (2017), coefficients α, β are scalars drawn from a monotonic function. In this work, we consider that a reasonable choice for such a monotonic function is the Shannon's entropy (Shannon, 1949; Charniak, 1996; Aizawa, 2003). The asymmetric model therefore takes into account language structure, which reflects in the resulting phrase embedding p.

Since the composition approach (1) is plain and natural, it has encouraged recent work. For instance, Tian et al. (2017) described theoretical conditions for the vector averaging operation as a model of composition in distributional semantics. Given word embeddings $x_{[\mathrm{DM}]}$ and $x_{[\mathrm{H}]}$ that are geometrically uncorrelated, then making $\alpha = \beta = \frac{1}{2}$ actually causes p approaches zero. In other words, the average operation causes embeddings of words co-occurring with low or moderated frequencies to cancel each other, so $p \to 0$. This effect suggests a linguistic intuition: uncorrelatedness between the meanings of words of a phrase occurs when the speaker constructs composite meanings. In contrast, word meanings co-occurring frequently (e.g. words co-occurring frequently, like "cell phone") tend to be merged as a unique meaning. This prevents the word embeddings involved from mutual annihilation, and their addition is normalized by the average operation. In this way, the average embedding p can be seen as an implicit embedding shared by $x_{[\mathrm{DM}]}$ and $x_{[\mathrm{H}]}$. From our experiments, we interpret that the aforementioned observations on the linguistic features encoded in phrase weights partly explain the low performance of the simple average or sum of word embeddings for representing sentences (Section 8).

4.2. The sparseness in neural language models

Neural language models have one characteristic in common: the sparseness of word co-occurrence statistics induces orthogonality in their weights used as word embeddings (Elman, 1991). For instance, in recent work on neural language models (Bengio et al., 2003; Mikolov et al., 2013a), binary sparse vectors are used for representing input words as categorical variables (one-hot encoded vectors). These vectors build a canonical basis for $\mathbb{R}^{|V|}$. This basis encodes the vocabulary as an orthonormal set

$$e = \{e_1, \ldots, e_{|V|}\} = \begin{pmatrix} 1 & 0 & \cdots & 0 \\ \vdots & \ddots & \ldots & \vdots \\ 0 & \cdots & 0 & 1 \end{pmatrix}$$

before training. During training, the items $e_i \in e$ are iteratively projected onto hidden weights of a neural network. The larger the statistical changes due to a given projection, the larger the gradient on that sample, and the more the originally random weight vector is rotated towards its associated orthogonal axis e_i. Once the network was trained by maximizing log probabilities, the projection layer of the model has learned weight matrices (embedding matrices) that map the items e_i co-occurring moderately to a vicinity centered at the average (word embedding) of these items in the embedding space.

Importantly, according to Raina et al. (2007), this learning procedure is known as *sparse encoding*. Ultimately, this is similar to LSA (Latent Semantic Analysis) but there are two main differences. The eigenvectors of LSA constitute an orthogonal basis and represent topics of documents composed by words. Instead, in neural language models (seen as sparse encoders), the resulting embeddings are not restricted to be orthogonal. They can be considered as an *overcomplete representation* or an *overcomplete basis* (Duffin and Schaeffer, 1952; Kovačević et al., 2008; Lewicki and Sejnowski, 2000). Each of its items represents the senses of a word modeled by a vector average of sliding contexts.

4.3. Merging

As an extension of the geometrical observations on compositionality and the overcomplete basis, we propose weighted series of word embeddings for sentence representation. Weighted series can be seen as the sparse decomposition of a sentence into basis vectors (Duffin and Schaeffer, 1952; Fourier, 1822), the *atoms* of an overcomplete representation.

Pennington et al. (2014) offers a proof of that the distribution of co-occurrences is bounded by a *generalized harmonic number* (i.e. a set of logarithmic distributions). Such a proof and the analysis presented in Sections 4.1 and 4.2 suggest that the angles between word embeddings are also logarithmically distributed as they are combined to transmit information. For us, such a distribution relates Shannon's entropy, co-occurrence statistics and the sparse encoding of the sentences of a corpus. This reasoning in turn suggests that a weighted series model admits the structure of asymmetrical compositionality. On the one hand, this is encoded in the weighted contribution of word embeddings representing composite meanings. On the other hand, this model also admits the weighted contribution of merged meanings whose constituent words share implicit embeddings. In turn, a series of word embeddings with sparse weights is built by two main subsets of an overcomplete basis (the word embeddings). The former subset contains uncorrelated word embeddings carrying specific information. The second subset contains correlated word embeddings carrying less information than the former.

5. The modules of our sentence representation method

5.1. The word embeddings module

In this section we describe the word embedding methods that WISSE employs as one of its modules. One of the most popular methods is termed Word2Vec (W2V) (Mikolov et al., 2013a), which is a neural language model inspired by ideas proposed previously (Baxter, 1998; Bengio et al., 2003; Hinton et al., 1986). More recently, such ideas have been also applied to learning representations as a sparse encoding problem (Raina et al., 2007). Word embedding methods have acquired popularity as applications of NLP acquire importance, and easy-to-use implementations of these methods are key for that. Therefore, other methods have been proposed and they are still growing in popularity, e.g. Glove (Pennington et al., 2014) and FastText (Bojanowski et al., 2016).

Word embedding methods share similar elements in general. Their aim is to train a non-parametric model whose parameters are in turn the weights of multiple linear discriminators. The model learns a set of binomials (discriminators) modeling the co-occurrence (and non-co-occurrence) of each word type along with other words within sliding windows (Rong, 2014). Once the model is trained, its parameters (the learned representations) are used as word embeddings.

5.1.1. Word2Vec

There are two possible neural architectures producing word embeddings with W2V: the Skip-gram and the CBoW. In this work, we used the Skip-gram neural architecture for training word embeddings (Mikolov et al., 2013b).

Let $V = \{w_1, \ldots, w_t, \ldots, w_{|V|}\}$ be the vocabulary of a context corpus $D = \{w_1, \ldots, w_i, \ldots, w_{|D|}\}$. The main idea behind Skip-gram is to classify all word occurrences w_i as they are instantiated as word types ($w_i = w_t$) through the corpus. Multiple target labels are simultaneously *self-taught* to each w_t embedding (Raina et al., 2007). Each target set is the vocabulary of a sliding window having r labels being the words around w_i. Thus, the target associated with w_i takes the form $c_i = \{w_{i-r}, \ldots, w_{i+r}\}w_i$. Therefore, the parameters of the distribution modeling the set $\{c_1, \ldots, c_i, \ldots, c_{|D|}\}$ must be *self-learned* by the Skip-gram classifier.

To enable the classifier to be self-taught, Word2Vec algorithm transforms the input unlabeled text into a labeled one first. This amounts to obtain a training set of the form $(w_1, c_1), \ldots, (w_i, c_i), \ldots, (w_{|D|}, c_{|D|})$. As Skip-gram is actually a neural network, its output layer can be written in terms of its hidden layers as follows:

$$P(c_i|w_i = w_t) = P(w_{i-r}, \ldots w_{i+r}|w_t) = \frac{\exp(x_t, \varphi_{c_i})}{\sum_{w \in V} \exp(x_t, \varphi_w)}. \quad (2)$$

The vectors $x_t \in \mathbb{R}^d$ are word embeddings of the word types w_t and they are actually the columns of the first hidden layer of the Skip-gram. The dimension of the word embeddings is d, which equals the number of hyperplanes

described by the first hidden layer. Each context vector $\varphi_{c_i} \in \mathbb{R}^d$ is the average of weight vectors of the hidden layer, which correspond to the $2|c_i|$ context words $\{w_{i-r}, \ldots, w_{i+r} | r = 1, 2, \ldots, |c_i|/2\} = c_i$ surrounding w_i. Rows of matrix $\Phi_V = \{\varphi_w \in \mathbb{R}^d | w \in V\}$ are the weight vectors of the output layer.

5.1.2. Dependency-based word embeddings (Dep2Vec)

One of the embedding methods we used as part of our experiments is an extension of W2V. The extension consists of a substantial modification of the notion of context window that uses dependency parsing as a surrogate for the standard co-occurrence window c_i in (2). Given a word w_i, dependency parsing is used for a establishing context restriction. The pair $(w_{i\pm 1,2,\ldots}, w_i)$ is considered to co-occur within the same context c_i whenever its words are related by a grammatical dependency. For example, according to the mentioned restriction, the co-occurrence of the words *John* and *sneezes* has higher possibilities than the co-occurrence of the words *house* and *sneezes*. *House* and *sneezes* can co-occur within a vicinity of words, but they do not hold a grammatical dependency. Therefore this pair is not considered as a co-occurring one. This is because of the object-subject distributional dependency restricting the actions that people and houses can do, i.e. people sneeze, but houses do not (Baroni and Lenci, 2010; Levy and Goldberg, 2014; Pagliardini et al., 2017).

5.1.3. FastText

The FastText model is another extension of W2V (Bojanowski et al., 2016). This model is a combination of the CBoW and Skip-gram Neural Network architectures. FastText is designed to segment the input text into contiguous n-grams of characters g_i. These contiguous n-grams are then grouped into context windows c_i. As in the case of W2V, c_i is associated with w_i. Thus, given a context window $c_i = \{g_1, \ldots, g_{|c_i|}\} w_i$ the self-taught model predicts a target window $c_i' = \{g_1, \ldots, g_{|c_i|}, w_i\}$, i.e. $P(c_i'|c_i) = P(g_1, \ldots, g_{|c_i|}, w_i | g_1, \ldots, g_{|c_i|})$. Notice that both c_i and c_i' are a bit different than the c_i built in Eq. (2). Furthermore, the target window contains the word w_i. This allows the model for learning different word embeddings for the word *as* and for the bigram ***as*** composing the word *where**as***. The same mechanism also allows FastText to infer out-of-vocabulary word embeddings.

5.1.4. Global vectors for word representation (Glove)

Global vectors for word representation (Glove) is a bit different method from most of its counterparts. Its main trait still holds. It can be considered as a self-taught method. By bringing back pioneer methods like LSA and Pointwise Mutual Information vectors (Jurafsky and Martin, 2009), this model transforms a sparse matrix $A \in \mathbb{R}^{|V| \times |V|}$ of word co-occurrences into co-occurrence probabilities (Pennington et al., 2014). The matrix A has entries a_{ij}, and each of them is function of the probability of that the word w_i co-occurs along with other words $w_{i\pm r}$ within a context window $c_i = \{w_{i\pm r} | r = 1, 2, \ldots, |c_i|/2\}$. The model learns the vector $x_i \in \mathbb{R}^d$ as the parameter of a regression function $g(x_i) = \langle x_i, \varphi_{c_i} \rangle$ on the non-zero log probabilities. The independent variable is $\varphi_{c_i} \in \mathbb{R}^d$, which is the average of the embeddings associated to words $w_{i\pm r} \in c_i$. Therefore, the regression is found by optimizing Eq. (3):

$$\mathcal{J}(x_i, \varphi_{c_i}) = \sum_{i,j}^{V} f(a_{ij}) (\langle x_i, \varphi_{c_i} \rangle - \log a_{ij})^2, \tag{3}$$

where x_i is taken as embedding of the word w_i. The weighting function $f(a_{ij}) = \left(\frac{a_{ij}}{\max\{a_{ij}\}}\right)^{\beta}$, with hyperparameter $\beta = 3/4$, allows $\mathcal{J}(\cdot, \cdot)$ to be adapted to the importance of co-occurrence probabilities a_{ij}.[6] Overall, the convex objective (3) is conceived to optimize the correspondence between the inner products $\langle x_i, \varphi_{c_i} \rangle$ and the log differences $\log a_{ij} = \log \frac{P(w_{i\pm r} = w_j | w_i)}{P(w_i)}$. Herein, w_i maps to the ith row of A, and w_j maps to its jth column.

5.2. The information-theoretic module

From an Information-Theoretic perspective, the TF–IDF transform identifies three subsets of words within a training corpus. The first one is the whole set of documents/sentences S (the Wikipedia in the case of this work). The second one is the subset of sentences S_i sharing a given word w_i of the corpus vocabulary V. And the third subset is the set of words

[6] Similar considerations are also referred to as "sub-sampling" in Mikolov et al. (2013a) and Bojanowski et al. (2016).

composing a given sentence s_j. These subsets comprise a structure such that TF−IDF, in fact (Aizawa, 2003), estimates the Mutual Information (MI) among its levels from the point of view of a given word w_i (i.e. $w_i \in s_j \subset S_i \subset S$).

Computing the MI between w_i and the rest of the set structure $s_j \subset S_i \subset S$ requires first to compute the entropy $H(s_j)$ of a sentence. Let s_j to have probability $P(s_j) = 1/N_S$ such that all $s_j \in S$ are supposed to be equally probable and uniformly distributed within a set of N_S sentences ($j = 1, 2, \ldots, N_S$). Therefore, $H(s_j)$ can be approximated as

$$H(S) = -\sum_{s_j \in S} P(s_j) \log P(s_j) \approx -\log \frac{1}{N_S}. \tag{4}$$

Next, the entropy of the corpus S given that w_i is observed is $H(S|w_i)$ and it can be approximated as:

$$H(S|w_i) = -\sum_{s_j \in S} P(s_j|w_i) \log P(s_j|w_i) \approx -\log \frac{1}{N_{w_i}}, \tag{5}$$

where N_{w_i} is the cardinality of S_i, the subset of sentences containing w_i. The entropy in (5) is defined in terms of the conditional probabilities $P(s_j|w_i)$ of observing sentences s_j given a particular word w_i.

According to Aizawa (2003), the MI, denoted as $I(V, S)$, of words $w_i \in V$ with respect to S needs to consider Eqs. (4) and (5) as follows in Eq. (6):

$$\begin{aligned} I(V,S) &= \textstyle\sum_{w_i \in V} P(w_i)[H(S) - H(S|w_i)] \\ &\approx \textstyle\sum_{w_i \in V} \sum_{s_j \in S} \frac{f_{ij}}{F} \log \frac{N_S}{N_{w_i}}. \end{aligned} \tag{6}$$

The probability of any w_i is computed as $P(w_i) = \sum_{s_j \in S} f_{ij}/F$, where f_{ij} is the frequency of w_i within a sentence s_j, and F is the sum of frequencies of all words in S. The MI between each word and the structure of sets of words $s_j \subset S_i \subset S$ can be obtained from each term of the summation (6). These MIs are given by the TF−IDF transform and we use them as weights of a series representing a sentence.

6. The proposed sentence representation model

In this paper, we propose using the Mutual Information between words and the set structure formed by sentences containing specific words and the whole corpus (either it is constituted by documents or by sentences) to weigh a series representing a sentence. Therefore, the terms of the series can be seen as a basis (or sparse encoding) spanning a *sentence meaning vector space*. The series has the general form

$$s(x, \varphi) = \sum_{w_i \in s} \varphi_i x_{w_i}. \tag{7}$$

In Eq. (7), the unknown coefficients φ_i weigh the contribution from each word embedding x_{w_i} to the sentence representation $s(x, \varphi)$. A natural instinct to determine the coefficients of the series is to optimize $\varphi_1, \ldots, \varphi_{|s|}$ without (or with few) assumptions about them. However, in this work we propose taking into account information features of each word obtained from a corpus as prior knowledge (Schölkopf et al., 1997). This prior knowledge relies on the fact that topical meaningful words (e.g., noun phrases and named entities) have relatively low probability of being used (Robertson, 2004), therefore they are informative. Conversely, non-topical words (e.g., prepositions and determinants) have a high probability of being used. Therefore, they are much less informative. Note that this model takes into consideration the relationship between syntactic features and how word information changes in sentence comprehension (De Marcken, 1999; Frank, 2013; Hale, 2006; Mitchell and Lapata, 2010; Pereira, 2000; Těšitělová, 1992).

Let us instantiate our information-theoretic module (Section 5.2):

$$I(V,S) \approx \sum_{w_i \in V} \sum_{s_j \in S} \frac{f_{ij}}{F} \log \frac{N_S}{N_{w_i}}. \tag{8}$$

Now we use the prior knowledge it represents about words of any corpus of sentences/documents. At this point, we suppose the module is already trained on S. The inner summation of (8) runs over unseen sentences we want to

represent. Therefore, from a different corpus S', we select an unseen sentence $s_j \in S'$, which in turn contains words $w_1, w_2, \ldots, w_{|s_j|} \in s_j$. The outer summation of the module runs over the words of the vocabulary. Thus, we have the IDF components of the weights of the series, and it represents the information (knowledge) learned from the particular sentence corpus S through its vocabulary V:

$$\frac{1}{F}\left(\log\frac{N_S}{N_{w_1}}, \ldots, \log\frac{N_S}{N_{w_{|V|}}}\right), \tag{9}$$

In vector (9), N_{w_i} is the number of sentences in S sharing each w_i, and N_S is the total number of sentences in S (Section 5.2).

Now, the outer summation in (8) runs over words of the given sentence $s_j \in S'$ and gives the scalar f_{ij} (the raw word frequency of w_i within s_j). Therefore, we represent the information-theoretic module of our sentence representations as the MI weights $\varphi_i \in \mathbb{R}$ defined as the product:

$$\varphi_i = \langle f_{ij}, F^{-1}\log\frac{N_S}{N_{w_i}}\rangle .$$

Last, if each word embedding $x_{w_i} \in \mathbb{R}^d$ of the series (7) is weighted by φ_i, we have:

$$s_j(x,\varphi) = \sum_{w_i \in s_j} \langle f_{ij}, F^{-1}\log\frac{N_S}{N_{w_i}}\rangle \; x_{w_i}, \tag{10}$$

which is the final sentence representation model (WISSE). See Fig. 1.

As an example, we show a sketch of the sentence *"The dog barks"*. Our model (10) allows to see how some $s(\cdot,\ \cdot)$ would look like geometrically (Fig. 2). The sentence sketch could have weights like $\varphi_{THe} = 0.075, \varphi_{dog} = 0.53, \varphi_{barks} = 0.37$. Thus, the example $s[(x_{The}, x_{dog}, x_{barks}), \varphi]$ can be represented as

$$\begin{aligned} s(x,\varphi) = s[(x_{The}, x_{dog}, x_{barks}), \varphi] &= \textstyle\sum_{w_i \in s} \varphi_i x_{w_i} \\ &= 0.075x_{The} + 0.53x_{dog} + 0.37x_{barks} \end{aligned}$$

Time and memory cost. Suppose we have an NLP system that relies on sentence representations. Once we have pre-trained word embeddings and MI weights, the sentence representations do not need to be explicitly inferred before they are needed. The representations can be computed *online* as the application requires them. this is possible due to *(i)* the modularity of our model and *(ii)* the low-cost operations needed to embed a sentence representation. Given a

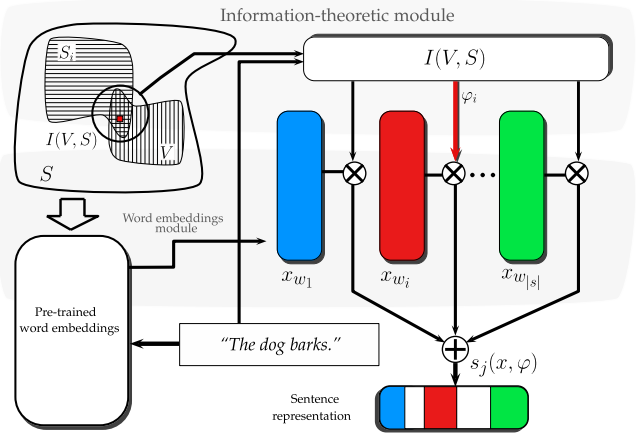


Fig. 1. Given a sentence (e.g. *"The dog barks."*), our model (10) asks to the word embedding module the word embeddings $\{x_{w_1}, \ldots, x_{w_{|s|}}\}$ associated with the words constituting such a sentence. In turn, the model asks to the information-theoretic module for the MI weights (TF–IDF) associated with each of the words os the sentence. These vectors are multiplied $\varphi_i = \langle f_{ij}, F^{-1}\log\frac{N_S}{N_{w_i}}\rangle$. According to our model, the sentence representations $s_j(\varphi, x)$ are obtained simply by summing the weighted word embeddings.

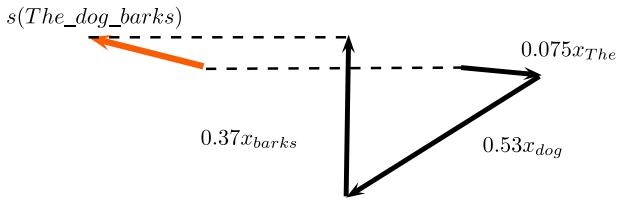


Fig. 2. A **hypothetical** sketch of the sentence vectors for the sentence *"The dog barks"* and the actual weights computed from the corpus. The vector projected by the dotted lines (left side) is the hypothetical summation of constituents: the sentence representation.

sentence of n words required by the NLP system, WISSE needs d scalar multiplications to weight each word embedding. WISSE also needs the summation of the n weighted word embeddings. Therefore, the time needed to embed each sentence representation is $T_s = T_{mul}(dn) + T_{sum}(nd) = T(2nd)$. Given that the average sentence length is very short ($n \in \{8, 12\}$ words), we see that $n \ll d$, so $T(2nd) \rightarrow O(kd)$, which is a linear bound to the dimension of the word embeddings. Regarding the memory requirements, the elements of the model can be indexed (e.g. by a database). In this case, for each sentence we only need to load n embeddings of d dimensions and the corresponding n weights, i.e. $n + nd = n(1 + d)$.

7. Design of experiments

In this section we present the multiple aspects considered in designing our experiments. The datasets we used are described, followed by a description of the set of hyperparameters that we used for model selection.

7.1. Datasets

We accomplished comparisons between our method and state-of-the-art methods on the basis of the SICK, the SemEval (2016) and the STS Benchmark (2017) datasets (Bentivogli et al., 2016; Agirre et al., 2016; Cer et al., 2017). Hyperparameter tuning of our method was performed on the train and development subsets of the STS Benchmark (2017). Also the trial subset of the SICK dataset was used to this end, so we considered all these subsets as a development dataset. Next, we used the following as test data: the train and test subsets of the SICK dataset, the test subset of the STS Benchmark, and the SemEval (2016) datasets.

The datasets used in this paper collect sentence pairs from multiple sources. These datasets contain texts written in English by Internet users, so they are linguistically varied and raise a number of interesting challenges for research on unsupervised sentence representation methods. An important part of the datasets comes from repositories like forums and news appearing daily. The datasets also consider digital dictionaries and knowledge bases (e.g. sense definitions from Wikipedia, WordNet, OntoNet, and sense contexts from FrameNet). In fact, the SICK dataset is a combination of carefully selected sentences from past years' SemEval STS datasets (2012−2014).

In all cases, the pairs of sentences are scored with an averaged semantic similarity gold standard manually annotated. The score ranges from 0.0, for unrelated pairs, to 5.0, for equivalent or literally equal pairs. The details about the collection protocol of the dataset are described elsewhere (Agirre et al., 2016; Bentivogli et al., 2016). We evaluated WISSE by taking advantage of this gold standard. A general summary of the datasets is shown below.

The Wikipedia dataset (general-purpose and unlabeled data). This dataset contains 5.11 million word types and it was downloaded from the Wikipedia dump (2012). We used this data to train FastText and Word2Vec word embeddings (the word embedding module of WISSE).

The STS Benchmark 2017. This dataset contains a mixture of sentence pairs from previous STS task up to 2018, the last year this task held. The dataset contains three subsets: train, development and test. The train data contains 1749 pairs of sentences, the development data contains 1500 pairs of sentences and the test data contains 1379 pairs of sentences.

The Answer-Answer dataset (SemEval-2016). This dataset contains 1572 pairs of answers which were extracted from the Stack Exchange Data Dump. The pairs of answers correspond to technical Stack Exchange forums, such as academia, cooking, coffee, DIY, and so on. This dataset is considered as a test set in our experiments.

The Headlines dataset (SemEval-2016). This dataset contains 1498 pair of news headlines collected from the Europe Media Monitor (EMM). This dataset is considered as a test set in our experiments.

The Postediting dataset (SemEval-2016). This dataset contains 3287 sentences from manually corrected machine translations of English-Spanish-French news. The translations were made using the Moses machine translation system. After that, the translations were manually post-edited so as to be corrected. This dataset is considered as a test set in our experiments.

The Plagiarism dataset (SemEval-2016). This dataset contains 1271 short answers to computer science questions that exhibit varying degrees of plagiarism with respect to Wikipedia articles. This dataset is considered as a test set in our experiments.

The Question-Question dataset (SemEval-2016). This dataset contains 1555 pairs of questions from the Stack Exchange Data Dump. The questions were extracted from technical Stack Exchange sites, such as academia, cooking, coffee, DIY, and so on. This dataset is considered as a test set in our experiments.

The SICK dataset (SemEval-2014). This dataset contains three subsets. The trial data contains 500 pairs of sentences and the train and test data sum up to 4906 pairs of sentences. All them are selected from English datasets used in previous STS tasks (2012-2014). The trial subset is considered as development data in our experiments. Both the train and test data are considered as a test dataset in our experiments.

7.2. Experimental setup

The initial preparation for our experiments was to train the word embedding algorithms. For both W2V[7] and FastText[8] word embedding models we used dimensions specified in Table 1, as well as their default training hyperparameters. The most important of them are the context window length (equal to 10) and the minimum word frequency (equal to 2). These word embedding models were trained on the Wikipedia dataset (Section 7.1). For Glove we used 50, 100, 200 and 300 dimensions pretrained embeddings available at the authors' website[9]. For Dep2Vec we used the 300–dimensional pretrained embeddings available at the authors' website[10]. All pretrained models were transformed into indexed versions to reduce memory usage and overhead, thus running multiple experiments much faster[11].

With respect to information weights, we trained multiple TF−IDF models on the Wikipedia for global weighting (Pedregosa et al., 2011). For local weighting, the STS dataset under observation was used to train its respective TF−IDF model. The main hyperparameters of the models' implementation were adopted as WISSE's weighting method hyperparameters (Table 1).

To get a reliable idea of the possibilities of WISSE, we performed a number of experiments using the datasets listed in Section 7.1 as development data. In this way, we tuned the hyperparameters of our model (WISSE), which are summarized in Table 1. Combinations of such hyperparameters result in different versions of our model. Thus the idea is to select the best models to predict semantic similarity on test data.

State-of-the-art sentence representation methods have already been evaluated in previous work over the test data we used here (Section 7.1). Thus, for each particular dataset we compared the Pearson's coefficient obtained by state-of-the-art methods with that obtained by the WISSE models selected. This allowed us to compare the behavior of these methods, through varied scenarios (Section 3).

[7] https://code.google.com/archive/p/word2vec.
[8] https://github.com/facebookresearch/fastText.
[9] https://nlp.stanford.edu/projects/glove.
[10] http://u.cs.biu.ac.il/~yogo/data/syntemb/deps.words.bz2.
[11] https://github.com/iarroyof/sentence_embedding.

Table 1
Hyperparameters tuned to study WISSE's performance.

Name	Description	Values
Dataset	The dataset for evaluation and hyperparameter tuning (see Section 7.1)	Evaluation: Plagiarism, Headlines, Answer-Answer, Question-Question, Postediting, STS Benchmark 2018 test (Bmk_test) and SICK train-test (Sick_trts). Development: STS Benchmark 2018 train and development (Bmk_train, Bmk_dev), and SICK trial (Sick_trial)
Embedding	Word embedding method	W2V, Glove, FastText and Dep2Vec
Size	The embedding dimensions	50d, 100d, 200d, 300d, 400d, 500d, 700d, 1000d.
Comb.	The combination method to obtain a unique embedding for a sentence	sum and avg. (embedding average)
Weighting method	Weighting method giving word embedding coefficients. All weighting schemes were optionally computed with stop words removal. To denote it, the suffixes -wst (with stop words†) and -ost (without stop words) were added. In the same way, the suffixes -bin, -freq and -log were added to denote whether the TF vector was computed as binary, with word frequencies and as the logarithm of frequencies (sublinear), respectively.	• TF−IDF global −weights learned from Wikipedia (glob-tfidf), • IDF global −idf learned from Wikipedia (glob-idf), • TF−IDF local −weights learned from the STS dataset under study (loc-tfidf), • IDF local −weights learned from the STS dataset under study (loc-idf), • All weights equal to 1.0 (unweighted: simple word embedding summation or average).
Distance (sim.)	The distance to measure similarity between sentence vectors	Cosine, Euclidean, Manhattan.

† The list of stop words used in our experiments can be verified here.

In addition to tuning hyperparameters, we decided to test three different similarity functions for computing semantic similarity scores between WISSE sentence representations: the cosine similarity, the Euclidean distance and the Manhattan distance. Given that the Pearson's coefficient is scale-invariant, it is not needed to transform the cosine similarities (which are in $[-1.0, 1.0]$) or Euclidean and Manhattan distances (which are in $[0.0, \infty)$) into the range of the gold standard (which is in [0.0,5.0]).

8. Experimental results

Our results are divided mainly into three parts. We present first the experiments made for finding the best baseline using simple word embedding summation and average. Second, we performed hyperparameter tuning of our information series model (Section 8.1). This resulted in four models that best performed on our development datasets. And third, we compared performances of the four selected models and state-of-the-art models on our test datasets.

To tune the hyperparameters of our model (WISSE), we used the SICK and the STS Benchmark 2018 to build development data as follows: from the SICK data, we used the train and test subsets, and from the STS Benchmark we used the trial and train subsets. To test the tuned models, we used the trial data from SICK, the test data from STS Benchmark, and the SemEval STS 2016 data.

8.1. Hyperparameter tuning (development)

We start by obtaining the best baselines. In Table 2, three sections (separated by double line) are shown corresponding to each development dataset we used (first column: Bmk_dev, Bmk_train, Sick_trial). For the three development datasets, FastText embeddings performed the best for simple summation and averaging. Glove appears in second place when sentence representations are semantically compared by using distances. Most of times, 300–dimensional embeddings (with stop words embedded, wst) resulted better baselines than other dimensions excepting the experiments on the Sick_trial dataset, where 200–dimensional embeddings worked better.

In Table 3, three sections (separated by double line) are shown corresponding to each development dataset we used (first column: Bmk_dev, Bmk_train, Sick_trial). The first three rows (limited by single line) of each section are the best development results considering global (glob) and local (loc) weights of the information series. The table shows that for the Bmk_dev dataset the highest correlation score (ρ) can be attained by using either global (glob) or local (loc) tfidf weights (using only idf weights did not help in any case). For this dataset, striping stop words (ost) was the best approach. The best similarity measure was the cosine, which is scale-invariant. Therefore, this measure is insensitive to the combination methods we used, i.e. vector average (avg) and vector summation (sum). Notice that this similarity is also the best fit for 300–dimensional FastText word embeddings. The unique difference between the best results with local and global weighting is how the TF vector was computed. For local weights, the

Table 2
Hyperparameter tuning of unweighted word embeddings (simple summation and simple average) for sentence representation on the STS Benchmark 2018 and SICK datasets (trial, train and dev.).

Dataset	Stop w.	Embedding	Size	Comb.	Sim.	Score (ρ)
Bmk_dev	wst	FastText	300d	sum	cosine	**0.72334**
	wst	FastText	300d	avg	cosine	**0.72334**
	wst	FastText	300d	sum	euclid.	0.44872
	wst	Glove	300d	avg	euclid.	0.62397
	wst	Glove	300d	sum	manha.	0.48289
	wst	Glove	300d	avg	manha.	0.62752
Bmk_train	wst	FastText	300d	sum	cosine	**0.71684**
	wst	FastText	300d	avg	cosine	**0.71684**
	wst	FastText	300d	sum	euclid.	0.34673
	wst	FastText	200d	avg	euclid.	0.64918
	wst	Glove	300d	sum	manha.	0.38620
	wst	Glove	300d	avg	manha.	0.64965
Sick_trial	wst	FastText	200d	sum	cosine	**0.70666**
	wst	FastText	200d	avg	cosine	**0.70666**
	wst	W2V	1000d	sum	euclid.	0.53579
	wst	FastText	500d	avg	euclid.	0.63400
	wst	Glove	300d	sum	manha.	0.55085
	wst	FastText	500d	avg	manha.	0.63436

Bold value indicates the best result for each dataset.

TF vector was computed as the logarithm of word frequencies (log), while for global weights the TF vector was computed as raw word frequencies (freq).

The behavior described for the three best results for the Bmk_dev dataset is similar to that observed for the Bmk_train dataset. This included the difference between using local and global weights. For local weights the logarithm of word frequency worked better, while for global weights raw word frequencies worked better. For this dataset global weights were benefit from stop words (wst), which did not occur for Bmk_dev dataset.

Table 3
WISSE hyperparameter tuning on the STS Benchmark 2018 and SICK datasets.

Dataset	Weighting method		Embed	Size	Comb.	Sim.	Score (ρ)
Bmk_dev	glob	tfidf-freq-ost	FastText	300d	sum	cosine	**0.77437**
	glob	tfidf-freq-ost	FastText	300d	avg	cosine	**0.77437**
	loc	tfidf-log-ost	FastText	300d	sum	cosine	0.77370
	loc	tfidf-log-ost	Glove	300d	sum	manha.	0.75018
	NA†	NA-NA-wst	FastText	300d	sum	cosine	0.72334
	loc	idf-log-ost	Glove	300d	avg	manha.	0.70247
	glob	tfidf-freq-ost	W2V	1000d	sum	euclid.	0.74206
	loc	idf-log-ost	Glove	300d	avg	euclid.	0.69825
Bmk_train	loc	tfidf-log-ost	FastText	1000d	sum	cosine	**0.73390**
	loc	tfidf-log-ost	FastText	1000d	avg	cosine	**0.73390**
	glob	tfidf-freq-wst	FastText	300d	sum	cosine	0.73350
	NA	NA-NA-wst	FastText	300d	sum	cosine	0.71684
	loc	tfidf-log-wst	Glove	300d	sum	manha.	0.70257
	loc	idf-log-wst	Glove	300d	avg	manha.	0.60705
	glob	tfidf-freq-wst	FastText	300d	sum	euclid.	0.70063
	loc	idf-log-wst	Glove	300d	avg	euclid.	0.60358
Sick_trial	glob	tfidf-freq-wst	FastText	200d	sum	cosine	**0.72144**
	glob	tfidf-freq-wst	FastText	200d	avg	cosine	**0.72144**
	loc	tfidf-freq-wst	FastText	200d	sum	cosine	0.71523
	NA	NA-NA-wst	FastText	200d	sum	cosine	0.70666
	glob	tfidf-freq-wst	W2V	500d	sum	euclid.	0.66980
	loc	idf-log-wst	FastText	200d	avg	euclid.	0.61366
	glob	tfidf-freq-wst	W2V	300d	sum	manha.	0.66863
	loc	idf-log-wst	FastText	200d	avg	manha.	0.61404

† NA = Not Applicable hyperparameter for baseline (unweighted) representations. Bold value indicates the best result for each dataset.

WISSE sentence representations showed a different behavior for the Sick_train dataset, which can be related partly with its length (only 500 samples). In this case, there was not difference between using logarithmic and raw frequency weights for local and global weighting. What is consistent with the other datasets is that global weights need raw word frequency TF vectors, which is also consistent with the Information-Theoretic view of TF−IDF.

All best results for the Sick_trial dataset required including information provided by stop words in the sentence representations. Unlike the cases of Bmk_dev and Bmk_train, sentence representations for this dataset needed 200–dimensional word embeddings. FastText word embeddings were also the best in this case. The difference in performance between local and global weighting for the Sick_trial dataset was a bit larger than for the other datasets.

We consider that in all cases a meaningful improvement will not be obtained by choosing either local or global weights whenever other hyperparameters are selected properly and the input data for local weighting is large enough (our results suggest that > 1000 can be needed to measure STS with local weights). Our development results indicate that logarithmic TF vectors work better for local weighting, while raw word frequency TF vectors work better for global weighting. Notice that this holds at least for the information that can be estimated from our development datasets according to their length: the Sick_trial contains 500 samples, which differs from the 1749 samples of the Bmk_train and the 1500 samples of the Bmk_dev.

In the case of cosine similarity, averaging and summing word embeddings to represent sentences held not difference because this similarity normalizes vector magnitudes. However, for Euclidean and Manhattan distances, comparing sentence representations given by embedding summation gave different correlation scores than for representations given by embedding average. This effect was observed in the three development datasets, and it is due to the geometrical nature of distances (the length of the line segment passing through two points). Furthermore, notice that in all cases using distances as similarity measures was outperformed considerably by cosine similarity. Nevertheless, in all cases where we used distances, summing word embeddings worked better than averaging them.

Regarding word embedding methods, FastText showed much better properties for sentence representation than the other methods we studied. Representations built via Glove embeddings were compared better by using Manhattan distance as similarity measure than by using the other measures, but these embeddings had low performance with respect to FastText ones. In some cases, Word2Vec (W2V) embeddings appeared in our ranking, but not among the best results. This occurred for the Bmk_dev and the Sick_trial datasets. It is needed to say that dependency-based embeddings (Dep2Vec) did not appeared in the first places for any dataset. Notice that in Table 3 we also included the best baseline (NA-NA-wst FastText) from Table 2 in order to compare unweighted representations with the combinations of hyperparameters that best performed on development data. These baselines performed better than weighted Glove and W2V representations in most cases.

The hyperparameters ranked in Table 3 led us to select the best WISSE models (see Table 4). The observed behavior of these models on the Bmk_dev and Bmk_train datasets was relatively consistent. Thus, we selected two models from these datasets to perform STS on test data. It is interesting to observe the performance of sentence representations resulting from the selected models as the dimensionality of the word embeddings changes. In Fig. 3 we plotted the semantic similarity correlation score as a function of the dimensionality of the word embeddings used to represent sentences. Each of the plots in the figure corresponds to each of the two development datasets the selected models were tuned on. The dimensionality was varied for the four models on both datasets.

From Fig. 3(a), we can see that the score between human judgments and WISSE predictions increases proportionally with respect to dimensionality as it goes from 50 to 300. The highest scores are attained at this latter dimension (300). After that point, the scores can be either within a vicinity of the maximum (0.77437) or considerably lower than it. The same pattern replicates for the Bmk_train, a completely different dataset (Fig. 3(b)). Notice that for the Bmk_dev data, the behaviors of the four models are completely correlated, although with a difference that slowly increases among them. A similar thing occurred for the Bmk_train data, but the differences among models are much less noticeable. Observe that this latter dataset is 250 samples larger than the Bmk_dev dataset, which can be a factor influencing the stability of the models with respect to changes in their hyperparameters.

To observe separately the behavior of sentence representations made from simple summation or simple average of word embeddings we performed a number of experiments summarized in Table 2. In a similar fashion as in the case of information series tuned in Table 3, we tested multiple hyperparameters (excluding weighting methods). Namely, it was interesting to characterize simple summation and averaging of word embeddings for sentence representation in terms of the embedding methods, dimensionality, similarity measure and stop words.

Table 4
Best WISSE models selected via the development datasets. Their Labels is used to identify each model during test experiments.

Dataset	Embedding	Weighting method				Dim.	Label
	FastText	glob/loc	Weight	TF	Stop w.		
Bmk_dev		glob	tfidf	freq	ost	300d	Glob-X
		loc	tfidf	log	ost	300d	Loc-X
Bmk_train		glob	tfidf	freq	wst	300d	Glob-Y
		loc	tfidf	log	ost	1000d	Loc-Y
Sick_trial		glob	tfidf	freq	wst	200d	–
		loc	tfidf	freq	wst	200d	–

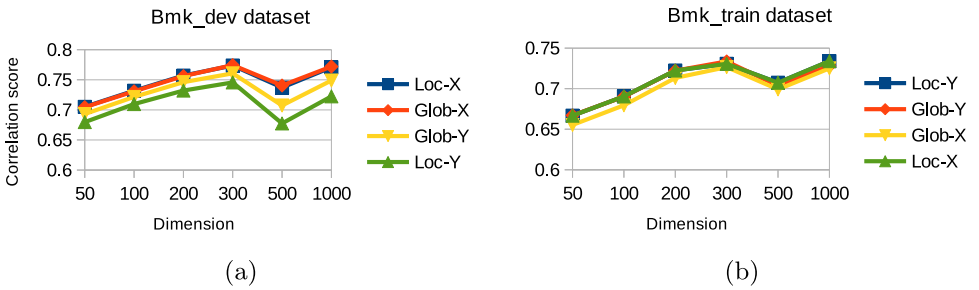


Fig. 3. Dimensionality Vs. correlation score on Bmk_dev and Bmk_train, two of our three development datasets.

8.2. Test results on the SICK dataset

In this subsection we compare the performance of our unsupervised sentence representation models selected from development experiments (Section 8.1) with the state-of-the-art methods in test experiments (Table 5). The selected WISSE models to test were labeled as glob-X, loc-X, glob-Y and loc-Y (Table 4).

One of the most popular tasks for evaluating unsupervised sentence representations is using them to predict Semantic Textual Similarity scores of the SICK dataset with the train and test subsets merged (thus we consider this merged dataset, train+test, as a whole test data). Therefore, we compared the correlation given by our best models and the correlation given by state-of-the-art models on such a dataset (see Table 5). Based on the Pearson's correlation coefficient, WISSE showed competitive performance with respect to all them.

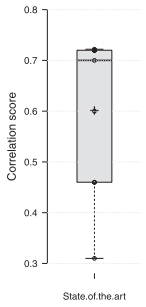
The state of the art gives a median score of $\rho = 0.700$. On the one hand, glob-X ($\rho = 0.712$) and loc-X ($\rho = 0.701$) ended within the upper quartile scores, which are $\rho \in [0.700, 0.720]$ (see the boxplot included in Table 5). On the other hand, glob-Y ($\rho = 0.724$) outperformed the methods considered as the state-of-the-art in this paper. Notice that the test dataset used in this subsection contains 4906 samples, which apparently did not change the quality of local estimates of word information with respect to what was observed with the development datasets (probably excepting the Sick_trial dataset, which is much smaller). CHARAGRAM-PHRASE ($\rho = 0.700$) reached the median performance, but other popular methods in the state of the art barely reached the mean performance (SkipThoughts, $\rho = 0.600$) or did not even reach the interquartile range (Doc2Vec, $\rho = 0.460$).

8.3. Test results on the SemEval Benchmark STS task

There exist state-of-the-art results reported for the Benchmark STS dataset on the SemEval wiki pages[12]. Several systems in the SemEval wiki are claimed to be unsupervised. This categorization was made in the sense of the STS task. Nonetheless, in the sense of building the sentence representations, these methods actually learned sentence representations in a supervised fashion (Conneau et al., 2017; Wieting et al., 2016; Yang et al., 2018). The main approach used by these methods is learning and transfer from similar tasks such as Natural Language Inference and

[12] http://ixa2.si.ehu.es/stswiki/index.php/STSbenchmark.

Table 5
Performance of sentence representation models on the SICK dataset (train+test).

Method (reference)	Corr. score (ρ)	Boxplot
WISSE (glob-Y, 300d)	**0.724**	
Glove+WR (Arora et al., 2017)	0.722	
Sent2vec (Pagliardini et al., 2017)	0.720	
FastSent (Hill et al., 2016)	0.720	
C-PHRASE (Pham et al., 2015)	0.720	
WISSE (glob-X, 300d)	0.712	
FastText (NA-NA-wst, 300d)	0.704	
WISSE (loc-X, 300d)	0.701	
CHAR-PHR (Wieting et al., 2016)	0.700	
WISSE (loc-Y, 1000d)	0.694	
Skip-thoughts (Kiros et al., 2015)	0.600	
SDAE (Hill et al., 2016)	0.460	
Doc2Vec (Le and Mikolov, 2014)	0.460	
SAE (Hill et al., 2016)	0.310	

Bold names denote our method and bold value denote the best score.

paraphrase prediction (Bowman et al., 2015). Other approaches reported on the SemEval wiki are semisupervised, also in the sense of building sentence embeddings, e.g. Glove + SIF (Arora et al., 2017), Paragram (Wieting et al., 2015). Thus, in the same manner than in Section 8.2, in this section we accordingly compare against WISSE uniquely those state-of-the-art methods that learned sentence representations in a fully unsupervised fashion (with no human annotation in the sense of learning sentence representations). See Table 6.

For the Benchmark STS dataset Sent2Vec outperformed WISSE and had the maximum correlation score. However, our method (WISSE, glob-Y configuration) was in the second place, and surpassed the upper whisker of the state-of-the-art boxplot ($\rho = 0.670$). WISSE (loc-X) ended in the third place ($\rho = 0.662$), and within the 3rd quartile $\rho \in [0.650, 0.670]$. The mean was equal to the median ($\rho = 0.650$), and WISSE (glob-X) reached this performance. Doc2Vec with DBOW configuration has been showed better performance than with the usual PV-DM configuration (the configuration tested for SICK, in Section 8.2). Nevertheless, this method (DBOW) had performance a bit lower ($\rho = 0.649$) than the mean of the state of the art reported here. C-PHRASE did not reach the lower whisker ($\rho = 0.640$). And finally, the loc-Y configuration of our method had the minimum performance ($\rho = 0.502$).

8.4. Test results on the SemEval STS task

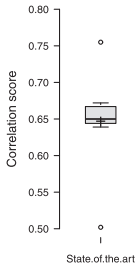
In this subsection, we compared WISSE with popular unsupervised sentence embedding methods from the state of the art on the SemEval STS 2016 (Agirre et al., 2016).[13]

We observed that, in general, WISSE (Loc-X, Loc-Y) was outperformed uniquely by Sent2Vec. Other state-of-the-art methods showed lower performances (Table 7).

An interesting result is that the state-of-the-art methods, including WISSE, found difficult to beat the BoW baseline in the Postediting dataset. Sent2Vec surpassed it by a small difference. Currently D2V (*Paragraph Vector*) and Skip-Thoughts are very popular methods, however their performances in the STS tasks reported in this paper were even lower than that of the baselines (BoW and word embedding average). For instance, the two highest mean performances were registered on the Plagiarism (0.7597) and the Postediting (0.7560) datasets. This suggests that these datasets represent less complex STS tasks than the other datasets. In this sense, while Sent2Vec and the four WISSE models surpassed the mean performances, Skip-Thoughts did not reach the first quartiles (0.6960 for Plagiarism, and 0.7530 for Postediting) and D2V barely reached the first quartile for the Postediting dataset. Finally, we noticed that the simple average of W2V word embeddings showed very similar performances with respect to those obtained with D2V, although the former outperformed the second in most of the cases and outperformed considerably

[13] This method was tested by the MayoNLP team (Afzal et al., 2016).

Table 6
Performance of sentence representation models on the Benchmark STS test dataset.

Method (reference)	Corr. score (ρ)	Boxplot
Sent2vec (Pagliardini et al., 2017)	**0.755**	
WISSE (glob-Y, 300d)	0.672	
WISSE (loc-X, 300d)	0.662	
WISSE (glob-X, 300d)	0.650	
Doc2Vec DBOW (Le and Mikolov, 2014)	0.649	
C-PHRASE (Pham et al., 2015)	0.639	
FastText (NA-NA-wst, 300d)	0.637	
WISSE (loc-Y, 1000d)	0.502	

Bold names denote our method and bold value denote the best score.

Skip-Thoughts in all cases. The FastText (NA-NA-wst) baseline had a similar, but higher, performance through the datasets than W2V (except by the Answer-Answer dataset).

9. Discussion

The differences in the behavior of our sentence representation method with respect to the textual properties of each dataset are meaningful. The property that mainly affected the performance of sentence representations in STS tasks was sentence length differences in pairs. That is, semantic similarity measuring suffered perturbations when the pairs were constituted by a long sentence and by a phrase (or very short sentence). This task is known as *cross-level semantic relatedness* (Jurgens et al., 2016). These kind of pairs are also found in textual entailment tasks, rather than in STS tasks.

Other observation related with text size came from the size differences of training corpus when TF–IDF weights are computed globally and locally. Our experiments on development data showed that globally estimating word information within a sentence is better when such estimation is consistent with Information Theory (i.e. word information comes from probability, which must be estimated from raw word frequency. See Section 5.2). Such consistency comes from the fact that the Information-Theoretic perspective mentioned describes sentences from a global point of view. Nevertheless, such point of view shows a limited structure of the text subsets constituting the corpus at this moment (i.e. words contained in documents that in turn are contained in the Wikipedia). We think that this limitation was partially compensated with logarithm for local weighting of our model. Therefore, the structure of subsets of text must to be studied in mode detail to be

Table 7
WISSE and state-of-the-art sentence embedding models on SemEval STS datasets.

Model	Ans.-Ans.	HDL	Plagiarism	Postediting	Ques.-Ques.
Sent2Vec	0.57739	0.75061	0.80068	0.82857	0.73035
WISSE (loc-X, 300d)	0.43576	0.70778	**0.81066**	0.78367	0.67033
WISSE (loc-Y, 1000d)	0.38093	0.70734	**0.81000**	0.79652	0.66567
WISSE (glob-X, 300d)	0.33919	0.68698	0.75976	0.75603	0.67497
WISSE (glob-Y, 300d)	0.38253	0.68374	0.78448	0.75306	0.70401
D2V (400d)	0.41123	0.69169	0.60488	0.75547	-0.02245
Skip-toughs	0.23199	0.49643	0.48636	0.17749	0.33446
FastText (NA-NA-wst)	0.37324	0.69559	0.75884	0.73573	0.62645
W2V (300d-average)	0.50311	0.66362	0.72347	0.73935	0.16586
Binary TF–IDF (BoW)	0.41133	0.54073	0.69601	0.82615	0.03844

Bold values indicates the best result for a dataset.

extended so as to learn word information estimates with better generalization ability (Vapnik, 1998). It is worth mentioning that the detail of this structure of sets was considered in certain degree by FastText embeddings as they are computed from character *n*–grams, which is actually the kind of analysis units in seminal work on structural theories of language (Chomsky and Schtzenberger, 1963).

The fact that WISSE outperformed the state of the art on the SICK dataset is interesting. This is because such a dataset constitutes a careful selection of sentence pairs coming from varied STS datasets (2012−2014). This selection included cross-level tasks, which are the most difficult tasks in existent STS tasks. We observed that it is very difficult to surpass the 0.720 score barrier and we think that it is mainly due to cross-level samples. In the case of the Benchmark and the STS datasets, WISSE was outperformed, which poses the challenge of determining what is affecting the generalization ability of our sentence representations in order to propose improvements. We think that this must be done by keeping simplicity and partial interpretability, which is one of the remarkable advantages of WISSE over all other methods.

We consider relevant that supervised sentence representation methods have been successful in learning both STS and cross-level tasks simultaneously by using computationally costly attention Neural Nets (Yin et al., 2016). WISSE can be methodologically compared with them in the sense of that attention weights learned by a Neural Net for STS play the same role than information weights WISSE learns using simply TF-IDF.

There are now 5 unsupervised sentence representation methods in the 0.720 barrier for the SICK data (Glove +WR, Sent2vec, FastSent and C-PHRASE). All these methods are completely based on neural networks. This supposes, in most cases, learning word embeddings first, and learning a transformation matrix the word embeddings are projected on after (as in the case of attention Neural Nets). According to results provided by our method, a weighted sum or series of word embeddings, we think learning additional transformation matrices can add unnecessary and exacerbated complexity to the problem of learning unsupervised sentence representations, at least for general semantic representation purposes. This leads to overparametrized models memorizing as much patterns as possible, which gives either very noisy and low-performance sentence representations or models being not a model, but almost a very big lookup table representation. This leads to implementation issues such as its need for unnecessary high computational power and its prohibitive use in simple NLP research/applications.

Our method (WISSE) integrates neural networks and Shannon's information. The former for learning word embeddings, and the second for weighting them. It is interesting that this weighting approach uses uniquely a scalar weight for each word embedding representing each word within a sentence. The weight of each word of the sentence controls the information provided by each embedding to the sentence representation. Given that this approach is much simpler (so has much lower computational cost and simple implementation) than purely neural network-based methods, we think it is much more useful for general purpose NLP applications and research.

It is important to say that our interest in this work was to represent sentences semantically. Most previous work include other kind of tasks like sentiment analysis, sentence classification and answer ranking for question-answering for evaluating sentences. Instead of that, we analyzed in more detail our model from a semantics point of view.

10. Possible improvements

Blinded information sources. Our current experiments only considered a basic version of our model using a three-level Shannon's information structure provided TF−IDF (i.e. words contained in documents that in turn are contained in the Wikipedia). Nonetheless, as we can see in Eqs. (4) and (5), the usual IDF weights make näve assumptions in terms of the probability measures of words and sentences. We believe that these assumptions could keep blinded important language structures (Lewicki and Sejnowski, 2000). For example, we did not estimate the actual probability measures of the stochastic processes underlying the different hierarchies of contexts. We expect that better sentence representations can be obtained by studying this issue in more detail.

Consider the case of *word order universals* (Derbyshire, 1977), which is not taken into account explicitly or is blinded in the structure of subsets of text WISSE models at this moment. A speaker emitting a sentence like 3 generally emphasizes the agent (*Paula*) performing the action (*to strike*). Conversely, in sentences like 4, the speaker emphasizes the experiencer (*Nacho*). This is not a general rule, but it surely provides an information spectrum related to informativeness of noun order.

Table 8
TF–IDF weights for the adverb form *not* within two sentences.

Sentence	TF–IDF weights
The man jumping is not wearing a shirt.	"jumping" (0.6537), "wearing" (0.4753), "shirt" (0.5679), "not" (**0.1894**)
A girl is close to a boy whose face is not shown.	"girl" (0.4258), "close" (0.3415), "boy" (0.4339), "face" (0.3833), "not" (**0.1959**)

3. *Paula struck Nacho.*
4. *Nacho was beaten by Paula.*

Notice that self similarity and hierarchy of these patterns can be found also in more complex constructions, e.g., *Paula struck Nacho because his offense was inadmissible*. In this example, the noun phrase *his offense* in the subordinate clause is also emphasized by the speaker (with respect to the adjective *inadmissible*). Nonetheless, for the whole sentence this emphasis is hierarchically less important than that put onto the agent in the main clause (*Paula*).

Advantages and disadvantages. At the moment, WISSE does not capture properly specific information patterns, like negation adverbs. For instance, similar representations of the sentences 1 and 2 are built:

1. *Computers will not condemn humanity.*
2. *Computers will condemn humanity.*

This holds because the same adverb form (*not*) is used many times when the speaker needs to negate any given fact. Thus, from the point of view of the information embedding that WISSE uses at this moment, this word is almost not informative. We show in Table 8 different weights for the word *not* within two randomly picked sentences from our corpus:

In fact, negation is an open issue in STS research. Depending on the way we consider the TF (either binary or logarithmic or frequency), the weights will change. Nonetheless their entropy keeps relatively constant. A possible solution for this issue learning context embeddings of emphasis via Relational Pattern Classification (Takase et al., 2016), although it initially requires supervised learning.

Notice that we are clearly identifying specific linguistic items participating in our model. This is an advantage of WISSE over most sentence representation methods. It is possible to manipulate the contribution of specific linguistic items within sentences. For instance, we could be interested in detecting similarity of sentences with respect to a specific entity (a noun phrase). In this case, it is sufficient to reinforce the weights associated with the word embeddings representing such an entity (Badarinza et al., 2017). This identifiability is not easy to attain by other state-of-the-art purely neural sentence embedding methods.

11. Conclusions

Our method outperformed the usual baselines of sentence representation on well-known STS tasks (BoW and word embedding average). Furthermore, the evaluation on the SICK dataset showed that our method outperformed the state of the art methods by reaching a correlation of $\rho = 0.724$. This is encouraging because measuring semantic similarity on this dataset is a difficult task for unsupervised sentence representation methods. In fact, the barrier of 0.72 is difficult to overcome for them. WISSE also reached state-of-the-art performance on the well-known SemEval (2016) STS task.

Our experiments confirmed our hypothesis stating that it is possible to well represent sentences by using the link between the contexts learned by word embeddings and the amount of information of words within a sentence. We exploited the mentioned link to learn without supervision the weights of a series of word embeddings representing a sentence. Interestingly, such weights are simple scalars that allowed our model to reach state-of-the-art performance in difficult STS tasks at low computational cost.

Acknowledgments

We thank to: CONACyT (Grant 386128), *Laboratorio Universitario de Cómputo de Alto Rendimiento* (IIMAS–UNAM), *Sistemas Linux y Super Cómputo* (IINGEN–UNAM), The Computational Genomics Research Program (PGC/CCG–UNAM) and The Postgraduate Program in Computer Science and Engineering–UNAM. Also thanks to Sarah Elizabeth Campion (Qatar University) for proof reading.

References

Afzal, N., Wang, Y., Liu, H., 2016. Mayonlp at SemEval-2016 task 1: semantic textual similarity based on lexical semantic net and deep learning semantic model. In: Proceedings of the SemEval NAACL-HLT, pp. 674–679.

Agirre, E., Banea, C., Cer, D., Diab, M., Gonzalez-Agirre, A., Mihalcea, R., Rigau, G., Wiebe, J., 2016. SemEval-2016 task 1: semantic textual similarity, monolingual and cross-lingual evaluation. In: Proceedings of the SemEval, pp. 497–511.

Agirre, E., Diab, M., Cer, D., Gonzalez-Agirre, A., 2012. SemEval-2012 task 6: a pilot on semantic textual similarity. In: Proceedings of the First Joint Conference on Lexical and Computational Semantics-Volume 1: Proceedings of the main conference and the shared task, and Volume 2: Proceedings of the Sixth International Workshop on Semantic Evaluation. Association for Computational Linguistics, pp. 385–393.

Aizawa, A., 2003. An information-theoretic perspective of TF–IDF measures. Inf. Process. Manag. 39 (1), 45–65.

Arora, S., Liang, Y., Ma, T., 2017. A simple but tough-to-beat baseline for sentence embeddings. In: Proceedings of the International Conference on Learning Representations (ICLR).

Arroyo-Fernández, I., 2015. Learning kernels for semantic clustering: a deep approach. In: Proceedings of the NAACL-HLT 2015 Student Research Workshop (SRW), pp. 79–87.

Arroyo-Fernández, I., Meza Ruiz, I.V., 2017. LIPN-IIMAS at SemEval-2017 task 1: subword embeddings, attention recurrent neural networks and cross word alignment for semantic textual similarity. In: Proceedings of the Eleventh International Workshop on Semantic Evaluation (SemEval-2017). Vancouver, Canada, pp. 199–203.

Arroyo-Fernndez, I., Torres-Moreno, J.-M., Sierra, G., Cabrera-Diego, L.A., 2016. Automatic text summarization by non-topic relevance estimation. In: Proceedings of the Eighth International Joint Conference on Knowledge Discovery, Knowledge Engineering and Knowledge Management - Volume 1: KDIR, pp. 89–100. doi: 10.5220/0006053400890100.

Badarinza, I., Sterca, A.I., Ionescu, M., 2017. Syntactic indexes for text retrieval. Inf. Technol. Ind. 5, 24–28.

Baroni, M., Dinu, G., Kruszewski, G., 2014. Don't count, predict! a systematic comparison of context-counting vs. context-predicting semantic vectors. In: Proceedings of the ACL, vol. 1, pp. 238–247.

Baroni, M., Lenci, A., 2010. Distributional memory: a general framework for corpus-based semantics. Comput. Linguist. 36 (4), 673–721.

Baxter, J., 1998. Theoretical Models of Learning to Learn. Springer, Boston, MA, US, pp. 71–94.

Bengio, Y., Ducharme, R., Vincent, P., Jauvin, C., 2003. A neural probabilistic language model. J. Mach. Learn. Res. 3 (Feb), 1137–1155.

Bentivogli, L., Bernardi, R., Marelli, M., Menini, S., Baroni, M., Zamparelli, R., 2016. Sick through the semeval glasses. lesson learned from the evaluation of compositional distributional semantic models on full sentences through semantic relatedness and textual entailment. Lang. Resour. Eval. 50 (1), 95–124.

Bojanowski, P., Grave, E., Joulin, A., Mikolov, T., 2016. Enriching word vectors with subword information. arXiv:1607.04606.

Bowman, S.R., Angeli, G., Potts, C., Manning, C.D., 2015. A large annotated corpus for learning natural language inference. In: Proceedings of the Conference on Empirical Methods in Natural Language Processing (EMNLP). Association for Computational Linguistics (ACL).

Cer, D., Diab, M., Agirre, E., Lopez-Gazpio, I., Specia, L., 2017. SemEval-2017 task 1: semantic textual similarity multilingual and crosslingual focused evaluation. In: Proceedings of the Eleventh International Workshop on Semantic Evaluation (SemEval-2017). Vancouver, Canada, pp. 1–14.

Charniak, E., 1996. Statistical language learning. MIT Press.

Chen, T., Xu, R., He, Y., Wang, X., 2017. Improving sentiment analysis via sentence type classification using bilstm-crf and cnn. Expert Syst. Appl. 72, 221–230.

Chomsky, N., Schtzenberger, M., 1963. The algebraic theory of context-free languages*. In: Braffort, P., Hirschberg, D. (Eds.), Computer Programming and Formal Systems. Studies in Logic and the Foundations of Mathematics. 35, Elsevier, pp. 118–161.

Collobert, R., Weston, J., Bottou, L., Karlen, M., Kavukcuoglu, K., Kuksa, P., 2011. Natural language processing (almost) from scratch. J. Mach. Learn. Res. 12 (Aug), 2493–2537.

Conneau, A., Kiela, D., Schwenk, H., Barrault, L., Bordes, A., 2017. Supervised learning of universal sentence representations from natural language inference data. In: Proceedings of the 2017 Conference on Empirical Methods in Natural Language Processing, pp. 670–680.

De Boom, C., Van Canneyt, S., Demeester, T., Dhoedt, B., 2016. Representation learning for very short texts using weighted word embedding aggregation. Pattern Recognit. Lett. 80, 150–156.

De Marcken, C., 1999. On the unsupervised induction of phrase-structure grammars. Natural Language Processing Using Very Large Corpora. Springer, pp. 191–208.

Derbyshire, D.C., 1977. Word order universals and the existence of OVS languages. Linguist. Inq. 8 (3), 590–599.

Duffin, R.J., Schaeffer, A.C., 1952. A class of nonharmonic fourier series. Trans. Am. Math. Soc. 72 (2), 341–366.

Elman, J.L., 1991. Distributed representations, simple recurrent networks, and grammatical structure. Mach. Learn. 7 (2–3), 195–225.

Er, M.J., Zhang, Y., Wang, N., Pratama, M., 2016. Attention pooling-based convolutional neural network for sentence modelling. Inf. Sci. 373, 388–403.

Ferrero, J., Besacier, L., Schwab, D., Agnès, F., 2017. Compilig at SemEval-2017 task 1: cross-language plagiarism detection methods for semantic textual similarity. In: Proceedings of the Eleventh International Workshop on Semantic Evaluation (SemEval-2017). Vancouver, Canada, pp. 100–105.
Firth, J.R., 1957. Papers in Linguistics 1934–1951. Oxford University Press, London.
Fourier, J., 1822. Theorie analytique de la chaleur. Chez Firmin Didot, père et fils.
Frank, S.L., 2013. Uncertainty reduction as a measure of cognitive load in sentence comprehension. Topics Cognit. Sci. 5 (3), 475–494.
Hale, J., 2006. Uncertainty about the rest of the sentence. Cognit. Sci. 30 (4), 643–672.
Harris, Z.S., 1968. Mathematical Structures of Language. Wiley, New York, NY, USA.
Hatzivassiloglou, V., Klavans, J.L., Eskin, E., 1999. Detecting text similarity over short passages: Exploring linguistic feature combinations via machine learning. In: Proceedings of the 1999 Joint Sigdat Conference on Empirical Methods in Natural Language Processing and very Large Corpora, pp. 203–212.
Hill, F., Cho, K., Korhonen, A., 2016. Learning distributed representations of sentences from unlabelled data. In: Proceedings of NAACL-HLT, pp. 1367–1377.
Hinton, G., McClelland, J., Rumelhart, D., 1986. Distributed representations. Parallel Distributed Processing: Explorations in the Microstructure of Cognition, 1. MIT Press, pp. 77–109.
Jurafsky, D., Martin, J.H., 2009. Speech and language processing: an introduction to natural language processing, computational linguistics, and speech recognition. Pearson Prentice Hall, Upper Saddle River, N.J. 9780131873216 0131873210.
Jurgens, D., Pilehvar, M.T., Navigli, R., 2016. Cross level semantic similarity: an evaluation framework for universal measures of similarity. Lang. Resour. Eval. 50 (1), 5–33.
Kalchbrenner, N., Grefenstette, E., Blunsom, P., 2014. A convolutional neural network for modelling sentences. In: Proceedings of the Fifty-second Annual Meeting of the Association for Computational Linguistics.
Kintsch, W., Mangalath, P., 2011. The construction of meaning. Topics Cognit. Sci. 3 (2), 346–370.
Kiros, R., Zhu, Y., Salakhutdinov, R.R., Zemel, R., Urtasun, R., Torralba, A., Fidler, S., 2015. Skip-thought vectors. In: Proceedings of the Advances in Neural Information Processing Systems, pp. 3294–3302.
Kovačević, J., Chebira, A., et al., 2008. An introduction to frames. Found. Trends® Signal Process. 2 (1), 1–94.
Landauer, T.K., Foltz, P.W., Laham, D., 1998. An introduction to latent semantic analysis. Discourse Process. 25 (2–3), 259–284.
Le, Q., Mikolov, T., 2014. Distributed representations of sentences and documents. In: Proceedings of the Thirty-first International Conference on Machine Learning (ICML). Beijing, China, pp. 1188–1196.
Levy, O., Goldberg, Y., 2014. Dependency-based word embeddings. In: Proceedings of the ACL, 2, pp. 302–308.
Lewicki, M.S., Sejnowski, T.J., 2000. Learning overcomplete representations. Neural Comput. 12 (2), 337–365.
Manning, C.D., Raghavan, P., Schütze, H., 2009. An Introduction to Information Retrieval. Cambridge University Press, Cambridge, United Kingdom.
Martin, D.I., Berry, M.W., 2007. Mathematical foundations behind latent semantic analysis. In: Landauer, Thomas K., McNamara, Danielle S., Dennis, Simon, Kintsch, Walter (Eds.), Handbook of Latent Semantic Analysis. Lawrence Erlbaum Associates Publishers, Mahwah, NJ, US, pp. 35–36. (Chapter 2).
Meza-Ruiz, I., Riedel, S., 2009. Jointly identifying predicates, arguments and senses using markov logic. In: Proceedings of the Human Language Technologies: The 2009 Annual Conference of the North American Chapter of the Association for Computational Linguistics. Association for Computational Linguistics, pp. 155–163.
Mihalcea, R., Corley, C., Strapparava, C., et al., 2006. Corpus-based and knowledge-based measures of text semantic similarity. In: Proceedingsof the AAAI, vol. 6, pp. 775–780.
Mikolov, T., Chen, K., Corrado, G., Dean, J., 2013a. Efficient estimation of word representations in vector space. arXiv:1301.3781.
Mikolov, T., Sutskever, I., Chen, K., Corrado, G.S., Dean, J., 2013b. Distributed representations of words and phrases and their compositionality. In: Proceedings of the Advances in Neural Information Processing Systems, pp. 3111–3119.
Mitchell, J., Lapata, M., 2010. Composition in distributional models of semantics. Cognit. Sci. 34 (34), 1388–1429. Cognitive Science Society, ISSN: 1551–6709
Onan, A., Korukoğlu, S., Bulut, H., 2017. A hybrid ensemble pruning approach based on consensus clustering and multi-objective evolutionary algorithm for sentiment classification. Inf. Process. Manag. 53 (4), 814–833.
Pagliardini, M., Gupta, P., Jaggi, M., 2018. Unsupervised Learning of Sentence Embeddings Using Compositional n-Gram Features. In: Proceedings of the North American Chapter of the Association for Computational Linguistics: Human Language Technologies (NAACL–HLT 2018), vol. 1, pp. 528–540.
Pedregosa, F., Varoquaux, G., Gramfort, A., Michel, V., Thirion, B., Grisel, O., Blondel, M., Prettenhofer, P., Weiss, R., Dubourg, V., Vanderplas, J., Passos, A., Cournapeau, D., Brucher, M., Perrot, M., Duchesnay, E., 2011. Scikit-learn: machine learning in Python. J. Mach. Learn. Res. 12, 2825–2830.
Pennington, J., Socher, R., Manning, C.D., 2014. Glove: global vectors for word representation. In: Proceedings of the Empirical Methods in Natural Language Processing (EMNLP), pp. 1532–1543.
Pereira, F., 2000. Formal grammar and information theory: together again? Philos. Trans. R. Soc. London A Math. Phys. Engi. Sci. 358 (1769), 1239–1253.
Pham, N.T., Kruszewski, G., Lazaridou, A., Baroni, M., 2015. Jointly optimizing word representations for lexical and sentential tasks with the c-phrase model. In: Proceedings of the ACL, vol. 1, pp. 971–981.
Raina, R., Battle, A., Lee, H., Packer, B., Ng, A.Y., 2007. Self-taught learning: transfer learning from unlabeled data. In: Proceedings of the Twenty-forth International Conference on Machine Learning. ACM, pp. 759–766.
Robertson, S., 2004. Understanding inverse document frequency: on theoretical arguments for IDF. J. Doc. 60 (5), 503–520.

Rong, X., 2014. word2vec parameter learning explained. preprint arXiv: 1411.2738.
Salton, G., Buckley, C., 1988. Term-weighting approaches in automatic text retrieval. Inf. Process. Manag. 24 (5), 513–523.
Salton, G., Fox, E.A., Wu, H., 1983. Extended Boolean information retrieval. Commun. ACM 26 (11), 1022–1036.
Schölkopf, B., Simard, P., Smola, A., Vapnik, V., 1997. Prior knowledge in support vector kernels. In: Proceedings of the Tenth International Conference on Neural Information Processing Systems. Cambridge, MA, pp. 640–646.
Shannon, C.E., 1949. Communication theory of secrecy systems*. Bell Syst. Tech. J. 28 (4), 656–715.
Spärk Jones, K., 1972. A statistical interpretation of term specificity and its application in retrieval. J. Doc. 28 (1), 11–21.
Takase, S., Okazaki, N., Inui, K., 2016. Composing distributed representations of relational patterns. In: Proceedings of the Fifty-forth Annual Meeting of the Association for Computational Linguistics (Volume 1: Long Papers), vol. 1, pp. 2276–2286.
Těšitělová, M., 1992. Quantitative Linguistics. Linguistics and Literary Studies in Eastern Europe, Academia Publishing House of the Czechoslovak Academy of Sciences.
Tian, R., Okazaki, N., Inui, K., 2017. The mechanism of additive composition. Mach. Learn. 106 (7), 1083–1130.
Vapnik, V.N., 1998. Statistical Learning Theory. Wiley, New York, NY.
Wieting, J., Bansal, M., Gimpel, K., Livescu, K., 2016. Towards Universal Paraphrastic Sentence Embeddings. In: Proceedings of the 4th International Conference on Learning Representations (ICLR).
Wieting, J., Bansal, M., Gimpel, K., Livescu, K., 2016. Charagram: Embedding words and sentences via character n-grams. In: Proceedings of the 2016 Conference on Empirical Methods in Natural Language Processing. Austin, Texas, pp. 1504–1515.
Wieting, J., Gimpel, K., 2017. Revisiting recurrent networks for paraphrastic sentence embeddings. In: Proceedings of the Fifty-fifth Annual Meeting of the Association for Computational Linguistics (Volume 1: Long Papers), 1, pp. 2078–2088.
Yang, Y., Yuan, S., Cer, D., Kong, S.-Y., Constant, N., Pilar, P., Ge, H., Sung, Y.-h., Strope, B., Kurzweil, R., 2018. Learning semantic textual similarity from conversations. In: Proceedings of The Third Workshop on Representation Learning for NLP, pp. 164–174.
Yazdani, M., Popescu-Belis, A., 2013. Computing text semantic relatedness using the contents and links of a hypertext encyclopedia. Artif. Intell. 194, 176–202.
Yin, W., Schütze, H., 2015. Discriminative phrase embedding for paraphrase identification. In: Proceedings of the HLT-NAACL, pp. 1368–1373.
Yin, W., Schütze, H., Xiang, B., Zhou, B., 2016. ABCNN: attention-based convolutional neural network for modeling sentence pairs. Trans. Assoc. Comput. Linguist. 4, 259–272.
Yu, J., Xie, L., Xiao, X., Chng, E.S., 2018. Learning distributed sentence representations for story segmentation. Signal Processing 142, 403–411 0165-1684.
Zhang, Z., Ge, S.S., He, H., 2012. Mutual-reinforcement document summarization using embedded graph based sentence clustering for storytelling. Inf. Process. Manag. 48 (4), 767–778.