Natural Language Processing **Emanuele Giona**
A.Y. 2018/2019

Homework 2: Neural sense embeddings

Task description and base model overview

Sense embedding is a technique for language modeling and feature learning applied in order to obtain dense vectors for words senses, instead of relying on large sparse vectors as the ones produced via a one-hot enconding approach. The need for vectors representing the different senses of a word is driven by the high degree of polysemy that is proper of natural languages, which a single vector per word fails to address when such word is ambiguous.

Similarly to word vectors, sense embeddings can be built through neural approaches equivalent to Word2Vec [1], and an example of this technique is the one presented in SensEmbed*: Learning Sense Embeddings for Word and Relational Similarity* [2], which is the reference paper for this homework assignment. The authors implement a Word2Vec model in the *Continuous Bag of Words* (CBOW) variant, for which a held-out word has to be predicted based on a context window. The aforementioned model is then trained on a corpus which is automatically annotated with BabelNet [3] synsets representing word senses, being able to also exploit a large structured knowledge source additionally to raw text corpora.

Data preprocessing

The dataset used for this homework is *EuroSense* [4], a multilingual sense-annotated corpus built from European Parliament transcripts in a similar way as described in [2], in its high precision version.
During a preliminary analysis of the corpus, several annotations were found misplaced, *i.e.* in the sentence ”Madam President , I would just like to clarify something .”:

- ”madam” annotated with ”chairman, chairperson, chair, chairwoman, president” (bn:00017517n)
- ”just” annotated with ”care, like, wish” (bn:00084526v)
- ”.” with ”madame” (bn:00052638n).

In order to obtain a corpus with no such annotations and theoretically exclude a negative influence on the model's performance, a rather simple sense alignment routine has been used: for each annotation, a bag of lemmas is created out of all WordNet lemmas associated to the BabelNet synset, then a new anchor is searched for in the sentence as a word which has at least one WordNet lemma overlapping with the bag of lemmas previously found; if no such word can be found, the annotation is discarded. A more detailed description of this process can be found in Algorithm 1; this routine has been able to correctly align 1 055 954 annotations, while discarding 4 573 464 annotations (around 30%).
The dataset to be processed in a Word2Vec-like model is built by only extracting English sentences out of EuroSense, and replacing each annotated anchor in the *lemma_synset* format (*i.e.* like_bn:00084526v). This shall be referred to as the DATASET throughout the rest of this report.

Implementation

The vocabulary has been built by excluding all words with less than 5 occurrences and applying a subsapling equal to 10^{-4}; both these countermeasures are not applied for *lemma_synset* words, which have always been kept in the vocabulary.

A basic Word2Vec model and a *synset-aware* Word2Vec one have been implemented, but for parameter tuning the basic Word2Vec has been trained on a subset[1] of the DATASET. The parameter tuning procedure followed a grid search approach according to Table 1, and the best results have been chosen on the basis of the lowest loss values recorded, such values being 4.605 on the training set and 4.592 on the development set after 30 epochs. The most important parameter seem to be the embedding size, since even the worse scoring setting using the bigger size (64) yields lower loss values than any setting using the smaller size (32).

The *synset-aware* Word2Vec model directly addresses the concept that vectors for word senses in a given synset should cluster, forcing the model to learn them this way by adding an extra term in the loss function.

[1] training set: first 10 000 sentences; development set: subsequent 5 000 sentences after the training set ones

The loss function used in the basic Word2Vec model is a Noise Contrastive Estimation loss with 16 negative samples; in the *synset-aware* Word2Vec model, the loss function used is the same as before, plus an extra term that is computed by considering the weighted average of the cosine distance between vectors of the current label word and vectors of words with the same synset (see Equations 1 and 2).

$$\frac{1}{k} \cdot \sum_{i=0}^{k} weight_i \cdot cos\_dist(label, word_i) \tag{1}$$

1: k is the number of words sharing the same synset

$$\forall i \in [1, k] \; weight_i = \begin{cases} 0 & \textbf{if } word_i = 0 \\ 1 & \textbf{otherwise} \end{cases} \tag{2}$$

2: Weights used in Equation 1; the single cosine distance is zeroed out when it refers to the label vector and the PAD token vector

In order to prevent penalizing too much a single training step when it comes to synsets having a high number of word senses associated to it, the model is only fed a *fixed size sample* of the words sharing the same synset: being a sample, it contains a word sense exactly once, and over the different training epochs, each word sense has the same probability of being sampled. This should also balance out cases in which a synset is responsible for many word senses and, the opposite case, in which a synset is only responsible for one.

Evaluation

Despite the higher loss values[2], the synset-aware Word2Vec approach looks promising when considering the closest words (see Table 2): closest senses are comparable, but it clearly shows it is learning to cluster senses.
In order to better emphasize on this, a spatial representation like PCA[3]: the basic Word2Vec model, although close, scatters the vectors around the space; the *synset-aware* Word2Vec model instead is able to bring the vectors referring to the same senses really close; examples might be Belgique_bn:00009714n and Belgium_bn:00009714n, or Sverige_bn:00049181n and Sweden_bn:00049181n, when considering the query word France_bn:00036202n (see Figure 2).

The similarity set used for this evaluation is WordSimilarity-353, overcoming missing words with zero-valued vectors when performing the cosine similarity measure. In order to compare the cosine similarity score to the human annotators' one, the Spearman and the Pearson's coefficients have been used, obtaining the following scores:

- Basic Word2Vec: 0.023 (Spearman), 0.005 (Pearson)
- Synset-aware Word2Vec: 0.027 (Spearman), 0.007 (Pearson).

Final considerations

The Synset-aware Word2Vec approach looks promising and further explorations could be performed on the size of the word senses sample, or even on the metrics used when considering such features to improve the loss function. It would probably benefit more from a richer sense annotations set than the basic Word2Vec model.

[2] See Figure 1; training set: 2.11305 (vs 2.10171); development set: 2.16291 (vs 2.15217). Scores wrapped in parenthesis refer to a basic Word2Vec model

[3] Visualized via TensorBoard: https://www.tensorflow.org/tensorboard

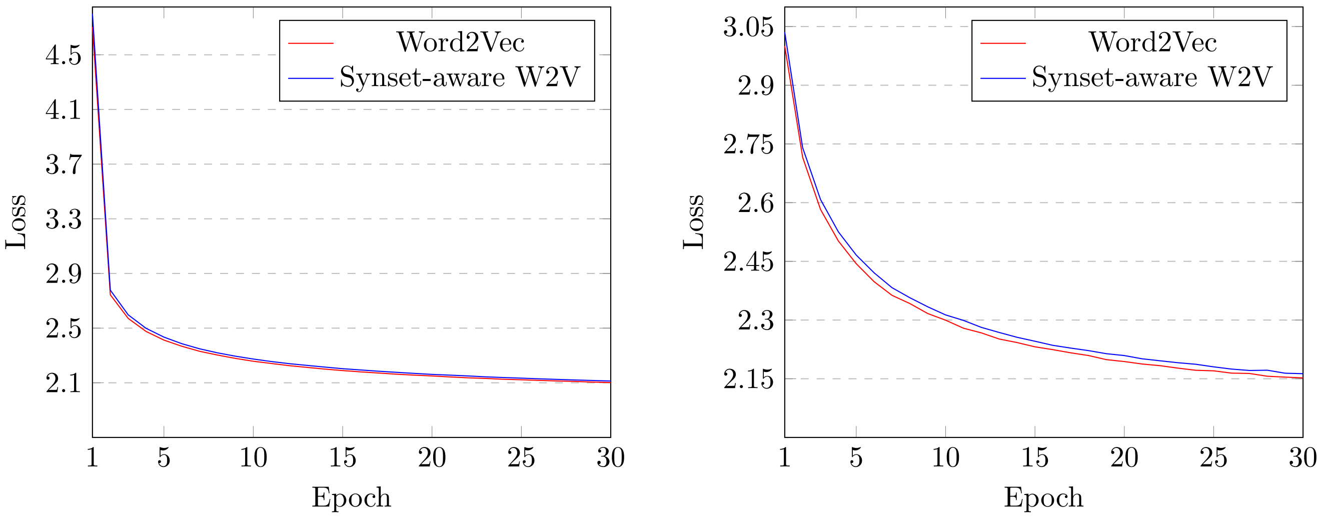
Tables and graphs

Algorithm 1 Sense alignment routine

```
$sentence \leftarrow$ sentence to be aligned
$annotations\_attributes \leftarrow$ attributes for each annotation (containing anchor and lemma)
$annotations \leftarrow$ annotations (BabelNet synsets)

$new\_ann\_attributes \leftarrow \emptyset$
$new\_annotations \leftarrow \emptyset$
foreach $attr \in annotations\_attributes, ann \in annotations$ do
    $all\_lemmas \leftarrow$ union of all lemmas associated to the WordNet synsets associated to $ann$
    $curr\_anchor \leftarrow attr[”anchor”]$
    $curr\_lemma \leftarrow attr[”lemma”]$

    if $curr\_lemma \notin all\_lemmas$ then
        $curr\_anchor \leftarrow$ ”UNDEFINED”
        $curr\_lemma \leftarrow$ ”UNDEFINED”

        foreach $word \in sentence$ do
            foreach $synset \in$ WordNetSynsets($word$) do
                $lemmas \leftarrow$ lemmas associated to $synset$
                if $all\_lemmas \cap lemmas \neq \emptyset$ then
                    $curr\_anchor \leftarrow word$
                    $curr\_lemma \leftarrow$ first element in $all\_lemmas \cap lemmas$

    if $curr\_anchor \neq$ ”UNDEFINED” then
        $attr[”anchor”] \leftarrow curr\_anchor$
        $attr[”lemma”] \leftarrow curr\_lemma$
        $new\_ann\_attributes \leftarrow new\_ann\_attributes \cup \{attr\}$
        $new\_annotations \leftarrow new\_annotations \cup \{ann\}$
```

Hyper-parameter	**Set of values**	**Best tuning**
Embedding size	32, 64	64
Learning rate	0.15, 0.10	0.15
Window size	2, 3	3

Table 1: Grid search used for hyper-parameter tuning and best scoring settings.



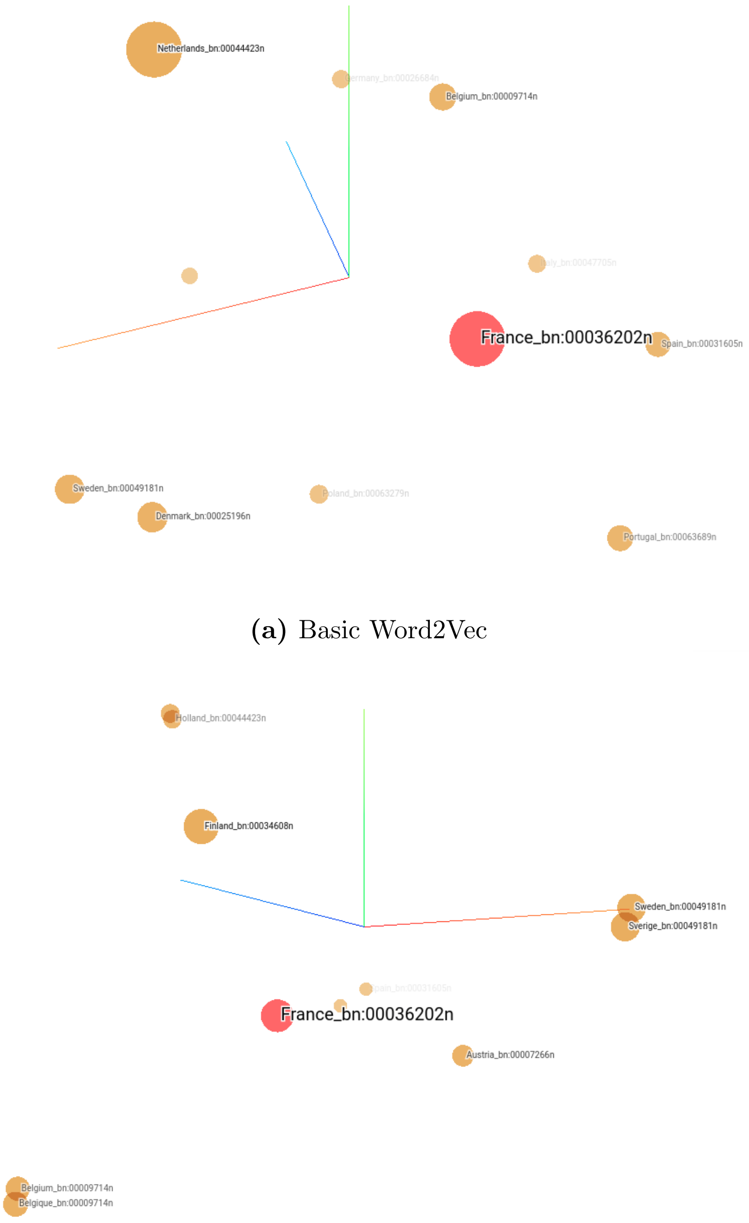
(a) Training set

(b) Development set

Figure 1: Loss value comparison after 30 epochs.

Query words	Basic Word2Vec	Synset-aware Word2Vec
Europe_bn:00031896n	Europe_bn:00021127n EU_bn:00021127n European_bn:00031898n world_bn:00029424n world_bn:00045153n	European_bn:00031898n European_bn:00102440a world_bn:00045153n Europe_bn:00021127n globe_bn:00029424n
France_bn:00036202n	Belgium_bn:00009714n Netherlands_bn:00044423n Austria_bn:00007266n Italy_bn:00047705n Spain_bn:00031605n	Austria_bn:00007266n Belgique_bn:00009714n Belgium_bn:00009714n Sverige_bn:00049181n Sweden_bn:00049181n
Council	council_bn:00023119n Bureau commission_bn:00021019n Councils summit	council_bn:00023119n Bureau Councils council_bn:00023121n Finance

Table 2: Top 5 closest senses, ordered.



(a) Basic Word2Vec

(b) Synset-aware Word2Vec

Figure 2: PCA visualization of the top 10 closest vectors to France_bn:00036202n

References

[1] Mikolov, Chen, Corrado, and Dean. (2013) *Efficient Estimation of Word Representations in Vector Space.* URL: https://arxiv.org/abs/1301.3781

[2] Iacobacci, Pilehvar and Navigli (2015). SensEmbed*: Learning Sense Embeddings for Word and Relational Similarity.* URL: https://www.aclweb.org/anthology/P15-1010

[3] Navigli and Ponzetto (2012). *BabelNet: The Automatic Construction, Evaluation and Application of a Wide-Coverage Multilingual Semantic Network.* URL: http://www.mt-archive.info/ACL-2010-Navigli.pdf

[4] Delli Bovi, Collados, Raganato, and Navigli (2017). *EuroSense: Automatic Harvesting of Multilingual Sense Annotations from Parallel Text.* URL: https://aclweb.org/anthology/papers/P/P17/P17-2094/