Text classification with semantically enriched word embeddings

N. PITTARAS

Institute of Informatics & Telecommunications, NCSR "Demokritos"
Patr. Gregoriou E & 27 Neapoleos Str, 15341 Agia Paraskevi, Greece
Department of Informatics and Telecommunications
National and Kapodistrian University of Athens
Panepistimiopolis, Ilisia, 15784, Athens, Greece

G. GIANNAKOPOULOS,

Institute of Informatics & Telecommunications, NCSR "Demokritos"
Patr. Gregoriou E & 27 Neapoleos Str, 15341 Agia Paraskevi, Greece

G. PAPADAKIS

Department of Informatics and Telecommunications
National and Kapodistrian University of Athens
Panepistimiopolis, Ilisia, 15784, Athens, Greece

V. KARKALETSIS

Institute of Informatics & Telecommunications, NCSR "Demokritos"
Patr. Gregoriou E & 27 Neapoleos Str, 15341 Agia Paraskevi, Greece

---

Abstract

The recent breakthroughs in deep neural architectures across multiple machine learning fields have led to the widespread use of deep neural models. These learners are often applied as black-box models that ignore or insufficiently utilize a wealth of pre-existing semantic information. In this study, we focus on the text classification task, investigating methods for augmenting the input to deep neural networks with semantic information. We extract semantics for the words in the preprocessed text from the WordNet semantic graph, in the form of weighted concept terms that form a semantic frequency vector. Concepts are selected via a variety of semantic disambiguation techniques, including a basic, a part-of-speech-based and a semantic embedding projection method. Additionally, we consider a weight propagation mechanism that exploits semantic relationships in the concept graph and conveys a spreading activation component. We enrich word2vec embeddings with the resulting semantic vector through concatenation or replacement and apply the semantically augmented word embeddings on the classification task via a deep neural network. Experimental results over established datasets demonstrate that our approach of semantic augmentation in the input space boosts classification performance significantly, with concatenation offering the best performance. We also note additional interesting findings produced by our approach regarding the behaviour of TF-IDF normalization on semantic vectors, along with the radical dimensionality reduction potential with negligible performance loss.

---

1 Introduction

The rise of deep learning has been accompanied by a paradigm shift in machine learning and intelligent systems. In Natural Language Processing applications, this has been expressed via the success of distributed representations (Hinton et al., 1984) for text data on machine learning tasks. These typically come in the form of *text embeddings*, which are vector space representations able to capture features beyond simple statistical properties, Such approaches try to evolve over the histogram-based accumulation used in methods like the bag-of-words model (Salton and Buckley, 1988). Instead of applying a hand-crafted rule, text embeddings learn a transformation of the elements in the input. This approach avoids the common problem of extreme feature sparsity and mitigates the curse of dimensionality that usually plagues shallow representations.

There have been numerous approaches to learning text embeddings. Early attempts produce shallow vector space features to represent text elements, such as words and documents, via histogram-based methods (Katz, 1987), (Salton and Buckley, 1988), (Joachims, 1998). Other approaches use topic modelling techniques, such as Latent Semantic Indexing (Deerwester et al., 1990) and Latent Dirichlet Allocation (Hingmire et al., 2013). In these cases, latent topics are inferred to form a new, efficient representation space for text. Regarding neural approaches, a neural language model applied on word sequences is used in (Bengio et al., 2003) to jointly learn word embeddings and the probability function of the input word collection. Later approaches utilize convolutional deep networks, such as the unified multi-task network in (Collobert and Weston, 2008), or introduce recurrent neural networks (RNNs), as in (Mikolov et al., 2011). Deep neural models are used to learn semantically-aware embeddings between words. (Mikolov et al., 2010; Mikolov et al., 2011) These embeddings try to maintain semantic relatedness between concepts, but also support meaningful algebraic operators between them. The popular word2vec embeddings (Mikolov et al., 2013a) learn such embedding spaces via Continuous Bag-Of-Words (CBOW) or skip-gram patterns, sometimes varying the context sampling approach (Mikolov et al., 2013b).

Despite numerous successful applications of text embeddings, most approaches largely ignore the rich semantic information that is often associated with the input data. Typically, such information – *e.g.* in the form of knowledge bases and semantic graphs – is readily available from human experts. This fact can function both as an advantage and a disadvantage in learning tasks. On the one hand, if the learned model is not restricted by human experts' rules and biases, it is free to discover potentially different (and often superior) intermediate representations of the input data (Bengio and Others, 2009) for a given task. On the other hand, ignoring the wealth of existing information means that any useful attribute is captured by relearning from scratch, a process that requires large amounts of training resources.

We claim that we need to investigate hybrid methods, combining the best of both worlds. As such, these methods will allow a model to search the feature space for optimal representations, while being able to exploit pre-existing expert features. We expect such a paradigm shift to affect a multitude of Natural Language Processing
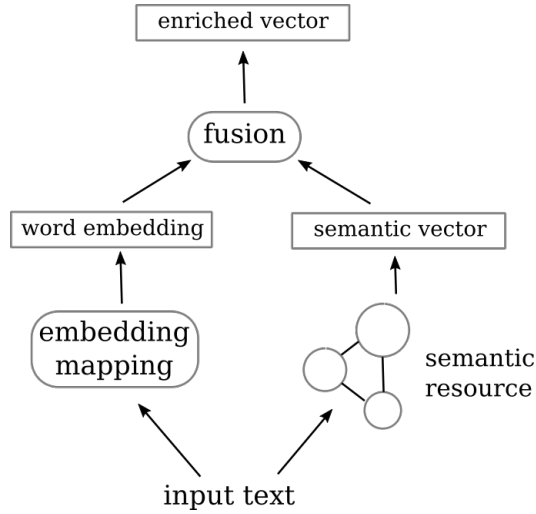
Fig. 1. Overview of our approach to semantically augmenting the classifier input vector.

tasks, from classification to clustering and summarization. Furthermore, it allows researchers to utilize a full range of different structure over resources: raw text, documents accompanied by meta-data and multimodal components (*e.g.*, embedded images, audio, *etc.*). However, we expect that the way of introducing semantic information to the model will affect training and the performance of the learned model on the task at hand.

In this context, this work suggests a method to integrate semantic information into a neural classification pipeline, and evaluates the outcome. We model the pipeline as a hybrid, branched architecture (cf. Figure 1), where each branch corresponds to one aspect of the hybrid: semantically-driven vs. raw-data-driven.

Given an input text, the semantically-driven branch, elaborated in Section 3, does the following:

- For each word it extracts semantic information from an appropriate resource of existing knowledge, such as a semantic graph.
- It generates a *semantic vector* for the word.
- It represents the whole text as a fusion of the word semantic vectors.

The raw-data-driven branch, utilizes raw data information to generate a word embedding as we find in many deep learning related works. Finally, we augment the word embedding output representations by the semantic vector, feeding the resulting enriched, hybrid representation to a deep neural network (DNN) classifier.

Based on this formulation, we address the following research questions in the context of the single-label text classification task:

1. Can semantic information increase the task performance, when applied in this setting? If so, how much?
2. How much do the different semantic disambiguation methods affect the above performance increase?

3. What is the effect of taking into account hypernymy relations in the semantic branch of the representation?

The rest of the paper is structured as follows. Section 2 surveys relevant works on semantic augmentation methods for classification problems, as well as the main techniques for enriching word embeddings with semantic information. In Section 3, we elaborate on our method, describing (a) the embeddings generation; (b) the semantic information extraction; and (c) the vector augmentation steps. Section 4 presents the experimental study that evaluates the performance of our workflow and compares it to the state-of-the-art in the field. It also addressing the above three research questions based on the results. We conclude the paper with a summary of our key findings along with directions for future work in Section 6.

2 Related work

In this study, we explore the introduction of semantic information into text-based embeddings, in the context of *single-label text classification.* Given a collection of text documents $T = \{t_1, \ldots, t_N\}$, a set of c predefined labels $L = \{l_1, \ldots, l_c\}$ and existing document-to-label annotations $G = \{(t_1, g_1), \ldots, (t_N, g_N)\}, g_i \in L$ such that the label of document t_i is g_i, the task is to find a classification function f that produces the correct label for each input document, *i.e.*, $f(t_i) = g_i$.

Specifically, we focus on DNN classifiers in conjunction with word embeddings for representing and feeding the input text to the predictive model.

In the literature, numerous studies leverage semantic knowledge to augment text mining tasks. For classification, graph-based semantic resources such as the Word-Net ontology (Miller, 1995) have been widely used to enrich textual information. Early approaches examined the effect of WordNet's semantic information on binary text classification, using rule-based discrimination (Scott and Matwin, 1998) as well as SVM classification (Mansuy and Hilderman, 2006). In (Elberrichi et al., 2008), the bag-of-words vector representation (Salton and Buckley, 1988) is combined with the WordNet semantic graph. A variety of semantic selection and combination strategies are explored, along with a supervised feature selection phase that is based on the chi-squared statistic. The experimental evaluation on the 20-Newsgroups and Reuters datasets shows that the semantic augmentation aids classification, especially when considering the most frequent related concept of a word. Frequency-based approaches are examined in (Nezreg et al., 2014) over the same two datasets, applying multiple classifiers to terms, WordNet concepts and their combination. The combined approach yields the best results for both datasets, however (a) it uses handcrafted features for the representation of textual information; (b) it employs shallow methods for classification and (c) it considers subsets of the two datasets.

On another line of research, neural methods are coupled with relationships from the WordNet semantic graph (Morin and Bengio, 2005), producing a language model where words are represented in a binary tree via hierarchical clustering – rather than deriving such a hierarchy from training data. The resulting model

trains faster and performs better than the bag-of-words baselines, but worse than the neural language model of (Bengio et al., 2003).

A bulk of later works modify the deep neural embedding training, with many of them investigating ways of introducing both *distributional* and *relational* information into word embeddings. Distributional information pertain to statistics from the context of a word, while relational information utilizes semantic relationships such as synonymy and hypernymy.

In more detail, the *"retrofitting" method* is used in (Faruqui et al., 2015) to shift word embeddings towards points in the embedding space that are more suitable to represent semantic relationships within an undirected graph. This is accomplished by post-processing the existing word vectors to balance their distance between their original fitted values and their semantic neighbours. The experimental analysis demonstrates the resulting improvements on the embeddings in a multilingual setting, with respect to a variety of semantic-content tasks. The retrofitting system is specialized in (Glavaš and Vulić, 2018) via a feed-forward DNN that explicitly maps semantic / relational constraints into modified training instances to produce specialized embeddings. The authors report significant gains in a series of tasks that range from word similarity to lexical simplification and dialog state tracking. The study in (Yu and Dredze, 2014) extends the neural language model of (Bengio et al., 2003) and (Mikolov et al., 2010) with semantic priors from WordNet and Paraphrase (Ganitkevitch et al., 2013). Their "Relation Constrained Model" (RCM) modifies the Continuous Bag-of-words (CBOW) (Mikolov et al., 2013b) algorithm, by modifying the objective function to consider only word pairs joined by a relation defined in the semantic ground truth. Additionally, they explore a joint model where the objective function considers a weighted linear combination of both corpus co-occurrence statistics and relatedness based on the knowledge resource. An experimental evaluation on language modeling, semantic similarity and human judgement prediction over a subset of the Gigaword corpus (Parker et al., 2011) demonstrates that using the joint model for pre-training RCM results in the largest performance increase, with respect to standard word2vec embeddings.

In (Fried and Duh, 2014), the authors model semantic relatedness by computing the length of the shortest path between WordNet synsets. This is mapped to a word pair by considering the maximum possible distance between candidate synset pairs associated with these words. A scaled version of this length regularizes the cosine similarity of the word pair corresponding word embeddings. Both distributional and semantic information are jointly trained via the ADMM multi-objective optimization approach (Boyd et al., 2011). The authors evaluate their graph distance measure – along with other WordNet distance approaches – on multiple tasks, such as knowledge base completion, relational similarity and dependency parsing. The overall results indicate that utilizing semantic resources provide a performance advantage, compared to using text-only methods.

In (Vulic and Mrkšic, 2018), embeddings are fine-tuned to respect the WordNet hypernymy hierarchy and a novel asymmetric similarity measure is proposed for comparing such representations. This results in state-of-the-art performance on multiple lexical entailment tasks.

Furthermore, the authors in (Xu et al., 2014) use skip-grams with relational, categorical as well as joint semantic biases in the objective function. To achieve this, they model a relation r between entities a and b as vector translation (*e.g.* $a + r = b$), *i.e.* modelling both entities and relations in the same vector space. Categorical knowledge is limited to fine-grained similarity scores, after discarding too generic entity relationships. All variants are evaluated on analogical reasoning, word similarity and topic prediction tasks, with the experimental results demonstrating that the joint model outperforms its single-channel counterparts (the semantically-aware networks and baseline skip-gram embeddings). In (Bian et al., 2014), the authors use external resources to introduce syntactic, morphological and semantic information into the generation of embeddings. Experimental results on analogical reasoning and word similarity sentence completion show that the semantic augmentation is the most reliable augmentation approach, compared to a CBOW baseline, with the other resources producing inconsistent effects on performance. The approach in (Luong et al., 2013) exploits morphological characteristics by training a recursive neural network at the level of a morpheme, rather than a word, which allows for generating embeddings for unseen words on-the-fly. The authors report large gains on the word similarity task across several datasets. Other approaches to fine-tuning embeddings seek to produce robust feature vectors with respect to language characteristics (Ruder et al., 2019), instead of enforcing explicit semantic relationships.

Some approaches apply their findings on text and/or document classification. The authors in (Card et al., 2018) propose a neural topic modelling framework that is able to incorporate metadata such as annotations / tags as well as document "covariates" (*e.g.*, year of publication), with tunable performance trade-offs. Experiments over a US immigration dataset show that this approach outperforms supervised LDA (Mcauliffe and Blei, 2008) on document classification. In (Li et al., 2017), the authors use a document-level embedding that is based on word2vec and concepts mined from knowledge bases. They evaluate their method on dataset splits from 20-Newsgroups and Reuters-21578, but this evaluation uses limited versions of the original datasets.

To tackle word polysemy and under/mis-representations of semantic relationships in the training text, many approaches build embeddings for semantic concepts (or "senses"), instead of words. Such "sense embedding" vectors are studied in (Chen et al., 2014), where the authors emphasize the weaknesses of distributional, cluster-based models like the ones in (Huang et al., 2012). Instead, they use skip-gram initialized word embeddings, aggregated to sense-level vectors by combining synset definition word vectors from WordNet. Word sense disambiguation (WSD) is performed via a context vector, with strategies based on word order or candidate sense set size, for each ambiguous word. Learning uses the skip-gram objective imbued with sense prediction. Evaluations on domain-specific data (*i.e.* small portion of the Reuters corpus) for coarse-grained semantic disambiguation (corresponding SemEval 2007 task (Navigli et al., 2007)) show that the proposed model performs similar to or above the state of the art, with the authors stressing the reusability of their approach.

In (Iacobacci et al., 2015), the authors introduce "Sensembed", a framework that generates both semantic annotation tags for a given dataset and sense-level embeddings from the resulting annotated corpus. In their approach, the BabelNet semantic graph (Navigli and Ponzetto, 2012) is used to annotate a Wikipedia dump so as to create a large semantically-annotated corpus via the BabelFly WSD algorithm (Moro et al., 2014). Out of the disambiguated corpus, sense vectors are produced with the CBOW model (Mikolov et al., 2013b). These vectors are subsequently evaluated in multiple word similarity, relatedness and word-context similarity tasks on multiple datasets. SensEmbed outperforms lexical word embeddings, as well as many related sense-based approaches on word similarity, with respect to Spearman correlation. It also performs better than mutual information-based baselines and word2vec embeddings on the SemEval-2012 relational similarity task (Jurgens et al., 2012). SensEmbed is used in (Bovi et al., 2015), where the knowledge base disambiguation and unification framework "KB-UNIFY" employs sense embeddings for the disambiguation stage, along with cross-resource entity linking and alignment, so as to unify and merge semantic resources. In (Flekova and Gurevych, 2016), the authors employ WordNet supersenses, *i.e.*, flat groupings of synsets, denoting synset high-level and more abstract semantic information that regular WordNet entities. BabelNet synsets are mapped to WordNet supersenses, using an automatically annotated Wikipedia corpus at multiple abstraction levels (Scozzafava et al., 2015). A range of evaluations on downstream classification tasks (subjectivity, metaphor, polarity classification) demonstrates that the proposed approach yields state of the art results, outperforming the exclusive use of distributional information. Further, the "AutoExtend" method (Rothe and Schütze, 2015) uses WordNet, considering words and synsets as a sum of their lexemes. Word embeddings are learned (or existing ones are modified) by a deep autoencoder, with the hidden layers representing synset vectors. Experiments on WSD, using the SensEval task corpora, show that AutoExtend achieves higher accuracy than an SVM-based approach with multiple engineered semantic features, with a subsequent combination of the two approaches further improving performance – indicating complementarity between them. In addition, an evaluation on word similarity shows that AutoExtend outperforms other systems ((Huang et al., 2012; Chen et al., 2014; Mikolov et al., 2013b)) as well as synset-level embeddings, in terms of Spearman's correlation. In (Goikoetxea et al., 2016), semantic embeddings are computed independently: a probabilistic random walk over the semantic graph outputs sequences of synsets, with the latter mapped to words via a dictionary of WordNet gloss relations. The resulting pseudo-corpus is fed to the skip-gram algorithm to learn semantic embeddings. Lexical and semantic vectors are subsequently combined in various ways, with simple concatenation outperforming more sophisticated semantic augmentation methods such as retrofitting, on similarity and relatedness datasets and tasks.

The approach in (Pilehvar et al., 2017) examines the effect of sense and supersense information on text classification and polarity detection tasks. Disambiguation is performed by mapping the input sentence into a subgraph of the semantic resource containing all semantic candidates per word. Then, the sense with the highest node degree is picked for each word, discarding the rest and pruning the subgraph ac-

cordingly. Additionally, supersenses are produced via averaging synset vectors with respect to the grouping of senses provided in WordNet lexicographer files. An experimental evaluation over the BBC, 20-Newsgroups and Ohsumed datasets shows that their approach introduces significant benefits in terms of F1-score, consistently improving the lexical embedding baseline on randomly initialized vectors. However, no improvement is observed when using pre-trained embeddings. For polarity detection, no consistent improvement is reported either. This is attributed to the short document sizes and the lack of word ambiguity in the examined datasets.

These approaches effectively introduce semantic information to deep neural architectures and word embeddings, but the evaluation of the refined embeddings on applied machine learning scenarios is limited, focusing for the most part on a variety of semantic similarity tasks or specialized, domain-specific classification tasks. Instead, this study focuses on a specific machine learning task, namely text classification, exploring the effect of semantic augmentation on deep neural models to the classification performance. Our worked is focused on the feature level, applying semantic enrichment on the input space of the classification process. We separate the embedding generation from the semantic enrichment phase, as in (Faruqui et al., 2015), where the semantic augmentation can be applied as a post-processing step. In fact, we model the semantic content as a separate representation of the input data that can be combined with a variety of embeddings, features and classifiers. Our approach extends earlier work on shallow features and learners (Elberrichi et al., 2008; Nezreg et al., 2014; Scott and Matwin, 1998; Mansuy and Hilderman, 2006) by augmenting deep embedding generators instead of local features. We also expand our investigation to additional semantic extraction and disambiguation approaches, by considering the effect of the n-th degree hypernymy relations and of several context semantic embedding methods. Finally, we expand and complement the findings of (Pilehvar et al., 2017), adopting multiple disambiguation schemes and a comparatively lower complexity architecture for classification.

3 Approach

We now delve into our approach for introducing external semantic information into the neural model. We present the textual (raw text) component of our learning pipeline in Section 3.1, the semantic component in Section 3.2 and the training process that builds the classification model in Section 3.3.

3.1 Text preprocessing and embedding generation

We begin by applying preprocessing to each document in order to discard noise and superfluous elements that are deemed irrelevant or even harmful for the task at hand. The processing tokenizes the original texts into a list of words and discards non-lexical elements such as punctuation, whitespace and stopwords[1]. To generate word embeddings, we employ the established word2vec algorithm (Mikolov

---

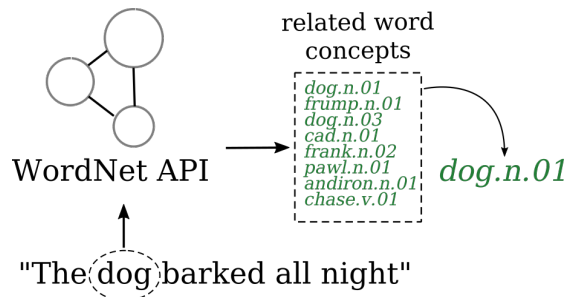[1] For stopwords removal, we use a popular list from (Porter, 2006).

Fig. 2. Example of the *basic* disambiguation strategy. Given the list of candidate synsets from the NLTK WordNet API, the first item is selected.

et al., 2013b). Specifically, we use the Continuous Bag-of-Words (CBOW) variant for the training process, which produces word vector representations by modelling co-occurrence statistics of each word based on its surrounding context. Instead of using pre-trained embeddings, we extract them from the given corpus, using a context window size of 10 words. To discard outliers, we also apply a filtering phase, which discards words that fail to appear at least twice in the training data. We train the embedding representation over 50 epochs (*i.e.*, iterations over the corpus), producing 50-dimensional vector representations for each word in the resulting dataset vocabulary. These embeddings represent the textual / lexical information of our classification pipeline.

3.2 Semantic enrichment

We now elaborate on the core of our approach, which infuses the trained embeddings with semantic information. First, we describe the semantic resource we use, WordNet. Then we introduce the semantic disambiguation phase which, given a word, selects a single element from a list of WordNet concepts as appropriate for the word. We continue with a description of the propagation mechanism we apply to spread semantic activation, *i.e.* to include more semantic information related to the concept in the word representation. We conclude with the fusion strategy by which we combine all information channels to a single, enriched representation.

3.2.1 Semantic resource

We use WordNet (Miller, 1995)[2], a popular lexical database for the English language that is widely used in classification and clustering tasks (Hung and Wermter, 2004; Morin and Bengio, 2005; Liu et al., 2007; Elberrichi et al., 2008). WordNet consists of a graph, where each node is a set of word senses (called synonymous sets or *synsets*) representing the same approximate meaning, with each sense also conveying part-of-speech information.

---

[2] See also `https://wordnet.princeton.edu/`

Synset nodes are connected to neighbours through a variety of relations of lexical and semantic nature (*e.g.*, *is-a* relations like hypernymy and hyponymy, part-of relations such as meronymy, and others). Note that while the terms "synset" and "concept" are similar enough to merit interchangeable use, we will use the word "concept" throughout the paper when not talking about the internal mechanics of WordNet.

Concerning WordNet synsets, in the following paragraphs we employ a notation inspired by (Navigli, 2009), as follows. In WordNet, the graph node encapsulating the everyday concept of a dog is a synset s, composed of individual word senses of near-identical semantic content, as below:

$$s = \left\{ dog_n^1, domestic\_dog_n^1, Canis\_familiaris_n^1 \right\}$$

The subscript of each word sense denotes POS information (*e.g.* nouns, in this example), while the superscript denotes a sense numeric identifier, differentiating between individual word senses. Given a word sense, we can unambiguously identify the corresponding synset, enabling us to resolve potential ambiguity (Martin and Jurafsky, 2009; Navigli, 2009) of polysemous words. Thus, in the following paragraphs we will use the notation $l.p.i$ to refer to the synset that contains the i-th word sense of the lexicalization l that is of a part-of-speech p.

For example, the common meaning of the word "dog" is approximately aligned with any of the word senses in the synset s, which in WordNet is accompanied by a definition: *"a member of the genus Canis (probably descended from the common wolf that has been domesticated by man since prehistoric times; occurs in many breeds"*. However, additional, more obscure senses of "dog" can be found in WordNet – *e.g.* dog_n^3, mapped to the synset defined as "informal term for a man"[3].

Thus, synset s will be referred to as $dog.n.01$, directly pointing to the (first) word sense of the word noun "dog"[4]. Using this notation, we move on to describe the disambiguation process.

3.2.2 Disambiguation

We extract information from WordNet via the NLTK interface [5]. Its API supports the retrieval of a collection of synsets as possible semantic candidates for an input word. It also allows traversal of the WordNet graph via the synset relation links mentioned above. Below we will denote string literals with a quoted block of text (*e.g.* "dog").

To select the most relevant synset from the acquired response of the API, we employ one of the following three disambiguation strategies:

1. The *basic* disambiguation strategy serves as a baseline, simply selecting the first synset from the retrieved list, discarding the rest. The NLTK WordNet

---

[3] As in phrases like *"You lucky dog!"*.

[4] This notation is also the one employed by the WordNet interface used in our semantic extraction process, the description of which follows.
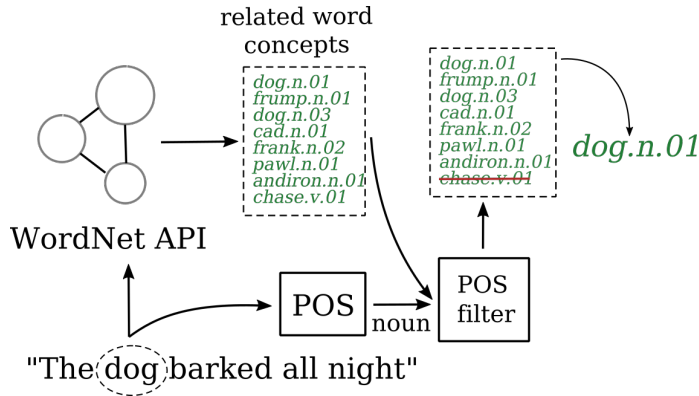
[5] https://www.nltk.org/data.html

Fig. 3. Example of the *POS* disambiguation strategy. Given the candidate synsets retrieved from the NLTK WordNet API, the ones that are annotated with a part-of-speech tag that does not match the respective tag of the query word are discarded. After this filtering process, the *basic* selection strategy is applied.

API ranks the retrieved synsets for a word query with respect to corresponding word sense frequencies computed in the SemCor sense-tagged corpus (Miller et al., 1993; Martin and Jurafsky, 2009; Navigli, 2009). Therefore, this method selects the most common meaning for the word (as computed in the SemCor text collection). An illustration of this selection process is shown in Figure 2.

2. The *POS* disambiguation strategy first filters the retrieved list of candidate synsets. The filtering discards all synsets that do not match the part-of-speech (POS) tag of the query word. Then the first remaining synset is selected.

   For example, when the word "can" is supplied with the POS tag "verb", the synset defined as "airtight sealed metal container for food or drink or paint etc." is discarded from the candidate synset list. After this filtering phase, the same mechanism as with the *basic* disambiguation process is applied. See Figure 3 for a visualization of this process.

3. The *context-embedding* disambiguation strategy uses a semantic embedding approach. For each candidate synset, related words are extracted from the accompanying example sentences together with the corresponding synset definition (*i.e.* the *gloss*).[6] Given the set of words from both sources, we compute the embedding mapping via the process described in Section 3.1. This associates every candidate synset with the set of embeddings of all words in its context.

   To arrive at a single vector representation for every candidate synset, this strategy averages all components across word embeddings in the context. This process maps the semantic information into a shared representation with the

---

[6] For instance, the *example sentence* for the synset *dog* is "*The dog barked all night*", while its *definition* was quoted in Section 3.2.1. Note that we consider synset definition sentences in addition to the WordNet examples, because around 70% of all synsets are associated with a single example.

lexical / textual one. In other words, it projects synsets into the same vector space. As a result, disambiguation by vector comparison is enabled: given a word w, its embedding e_w and a set of synset embeddings $S = \{e_1, \ldots, e_{|S|}\}$, we assign w to the synset $s = \underset{S}{\operatorname{argmin}}\,[dist(e_w, e_s)]$, where e_s is the aggregated synset embedding for synset $s \in S$ and $dist(\cdot, \cdot)$ denotes a vector distance metric.

To ensure adequate word context for generating representative semantic embeddings, we discard all synsets with fewer than 25 context words. The synset vector computation process from the whole WordNet, which is illustrated in Figure 4, results in 753 adequately represented synsets. The disambiguation process itself is depicted in Figure 5.

Overall, the *context-embedding* disambiguation strategy performs synset selection in a significantly more complicated manner than the other two strategies. Rather than using low-level lexical information (*basic* strategy) or lexical and syntactic features (*POS* strategy), this approach exploits the available distributional information in WordNet in order to match the input word to a synset.

This strategy bears some resemblance to other embedding-based disambiguation methods in the literature. However, given (a) the paper's focus on the downstream task of classification and (b) the multiple other disambiguation strategies examined, we decided to build a straightforward approach as described above; this way, we managed to reduce both the number of decision points (thus largely avoiding heuristics engineering and metaparameter optimization) as well as the computational requirements of this embedding-based disambiguation approach, in favor of a more robust, readily applicable algorithm. Comparatively, an example of a similar embedding-based approach is the work in (Chen et al., 2014), where the authors build synset embeddings via a process that includes averaging vectors of words that are related to WordNet synsets. However, their method is considerably more intricate compared to ours, since in their approach (i) semantic vectors are additionally fitted, with the aforementioned scheme being used just for sense vector initialization, (ii) a filtering step is used to process the WordNet gloss text prior to vector initialization, using a predefined subset of POS tags and iii) only words with representations close to the candidate word in a embedding space are considered, using a distance / similarity threshold. In contrast, *context-embedding* directly pools all available textual resources that accompany a synset in order to construct an embedding, *i.e.* utilizing all available distributional information WordNet has to offer. Additionally, no further fitting is applied to the resulting embedding, but it is used as-is in the downstream classification task for disambiguation purposes (*i.e.* to retrieve the synset that will be used in the actual semantic information extraction component).
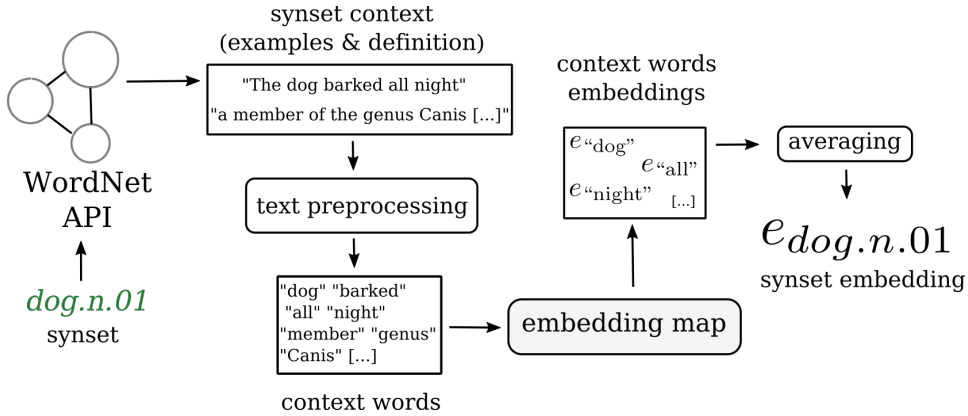
Fig. 4. Example of the synset vector generation for *context-embedding* disambiguation strategy. The context of each synset is tokenized into words, with each word mapped to a vector representation via the learned embedding matrix. The synset vector is the centroid produced by averaging all context word embeddings.
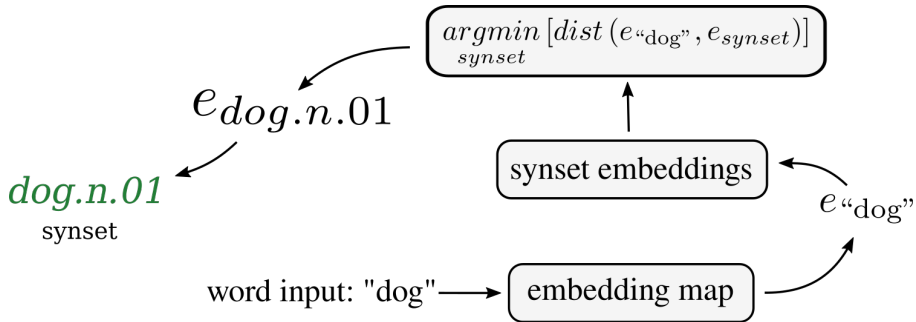
Fig. 5. Example of the disambiguation phase of the *context-embedding* disambiguation strategy. A candidate word is mapped to its embedding representation and compared to the list of available synset vectors. The synset with the vector representation closest to the word embedding is selected.

3.2.3 n−level hypernymy propagation

Since WordNet represents a graph of interconnected synsets, we can exploit meaningful semantic connections to activate relevant neighbouring synsets among the candidate ones. In fact, our approach propagates activations further than the immediate neighbours of the retrieved candidate synsets, to a multi-step, n-level relation set. This way, a spreading activation step (Collins and Loftus, 1975) propagates the semantic synset activation towards synsets connected with hypernymy relations with the initial match. In other words, it follows the edges labelled with *is-a* relations to include the encountered synsets in the pool of retrieved synsets.

The synsets extracted with this process are annotated with weights inversely proportional to the distance of the hypernymy level from the original synset. This

weight decay is applied to diminish the contribution of general and/or abstract synsets, which are expected to be encountered frequently, thus saturating the final semantic vector.

After a hyper-parameter tuning phase, we arrived at the configuration of a 3-level propagation process, with each level associated with a weight decay factor of 0.6. This mechanism enables words to share semantic information even if they do not belong to the same synset directly, but their mapped synsets can be linked via a short walk in the semantic graph – the longer the path, the lower the resulting relatedness weight.

To illustrate the spreading activation mechanism, consider querying the NLTK WordNet interface with the input word "dog" as an example. Using the *basic* disambiguation strategy (cf. Section 3.2.2), the first synset is selected out of the retrieved list – *i.e.* is the synset $dog.n.01 = \left\{dog_n^1, domestic\_dog_n^1, Canis\_familiaris_n^1\right\}$ and is valued with a unit weight. Subsequently, our spreading activation procedure is activated and operates as follows:

- The first step yields the direct hypernyms of the synset $dog.n.01$ in the Word-Net graph: $h_1 = \{x|dog.n.01 \text{ is-a } x\} = \{canine.n.02, domestic\_animal.n.01\}$, with $canine = \left\{canine_n^2, canid_n^2\right\}$ and $domestic\_animal.n.01 = \{domestic\_animal_n^1, domesticated\_animal_n^1\}$. These two synsets are thus assigned a weight of $0.6^1 = 0.6$.
- Next, we retrieve the hypernyms of each synset in h_1. This yields the synsets $h_2 = \{carnivore.n.01, animal.n.01\}$, each weighted with $0.6^2 = 0.36$.
- Finally, the third step produces the synsets $h_3 = \{placental.n.01, organism.n.01\}$, each with a weight of $0.6^3 = 0.216$.

As a result, the result of the 3-level spreading activation procedure on the word "dog" is a semantic vector with values: $[1, 0.6, 0.6, 0.36, 0.36, 0.216, 0.216]$ corresponding to weights for the synsets $[dog.n.01, canine.n.02, domestic\_animal.n.01, carnivore.n.01, animal.n.01, placental.n.01, organism.n.01]$. This example is also illustrated in Figure 6. Having this mapping of word to name-value collections, the next section describes the procedure by which the word-level information is fused to arrive at document-level vectors.

3.2.4 Fusion

As explained above, each semantic extraction process yields a set of concept-weight pairs for each word in the document. We want a single, constant length, semantic vector for each document. Thus, we form this vector by following a bag-of-synsets/concepts approach: we create a vector space where each dimension is mapped to one of the concepts discovered in the corpus; then we apply the semantic extraction to all documents in the corpus, mapping each of these documents in the space based on the frequency of a concept in the document. Thus, similar to the bag-of-word paradigms, we can generate two different vector types:

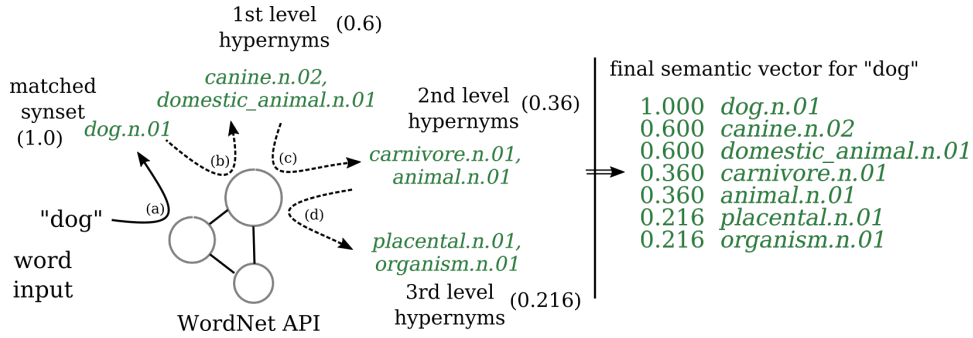1. We consider the raw concept frequencies over each document, arriving at

Fig. 6. Example of the spreading activation process for the input word "dog", executed for 3 levels with a decay factor of 0.6. The solid line (a) denotes the semantic disambiguation phase with one of the covered strategies. Dashed lines ((b) through (d)) represent the extraction of synsets linked with a hypernymy (*is-a*) relation to any synset in the source list. The numeric values represent the weight associated to synsets of each level, with the final semantic vector for the input word being listed to the right.

semantic vectors of the form $s^{(i)} = \{s_1, s_2, \ldots, s_d\}$, where $s_j^{(i)}$ denotes the frequency of the j-th concept in the i-th document. A concept in the semantic vector appears at least once in the training dataset. Note that concepts extracted from the test dataset, which do not appear in the training dataset are discarded.

2. We apply a weighting scheme similar to TF-IDF (Salton and Buckley, 1988) at the document and corpus levels, *i.e.*, by normalizing the document-level concept frequencies with the corresponding corpus-level frequencies: $w_j^{(i)} = s_j^{(i)}/\Sigma_{k \in [0,\ldots N]} s_j^{(k)}$, where $s_j^{(i)}$ stands for the raw frequency of the j-th concept in the i-th document, N is the number of documents in the dataset and $w^{(i)}$ is the final TF-IDF concept weight in the i-th document. This process reduces the importance of concepts that appear in too many documents in the corpus, similar to weight discounting of common words in a text retrieval setting.

After this post-processing stage, we are ready to incorporate the semantic vector into the classification pipeline. This is done in two ways:

1. The *concat* fusion strategy concatenates the embedding with the semantic vector, arriving at a semantically augmented representation that is fed to the downstream classifier.
2. The *replace* fusion strategy discards completely the lexical embedding, using only the semantic information for tackling the categorization task.

Since all cases above use semantic features that represent explicit concept-weight information (rather than explicitly distributed vectors), we do not fine-tune the augmented embeddings during training but keep the entire representation "frozen" to the original input values.

3.3 Training

We use a deep neural network (DNN) with 2 hidden layers, each containing 512 neurons. We arrived at this configuration after fine-tuning these two hyper-parameters through a grid search on a range of values: from 1 to 4 with a step of 1 for the number of hidden layers; values of $\{128, 256 \ldots, 2048\}$ for the number of neurons within each hidden layer. We use dropout with a heuristically selected 0.3 drop probability and position it after each dense layer to avoid overfitting. We train the DNN for 50 epochs over the total training data, with a 25-epoch early stopping, which allows the training to end prematurely, if the validation loss does not decrease for 25 consecutive epochs. We apply a 5-fold cross-validation split for training, classifying the input via a softmax layer. The learning rate is initialized to 0.1, applying a reduction schedule of a 0.1 decay factor every 10 epochs on loss stagnation.

3.4 Workflow summary

At this point we summarize the complete workflow of our approach to put everything we have described together, under a common view:

1. Preprocessing transforms each document into a sequence of informative words.
2. word2vec word embeddings are learned from scratch on these word sequences.
3. Semantic information for each document is extracted via the NLTK WordNet interface in the form of frequency-based concept vectors. This entails:
    (a) Concept extraction for each word by one of the disambiguation strategies (*basic*, *POS* or *context-embedding*).
    (b) 3-level hypernymy activation propagation based on the WordNet graph.
    (c) Raw-frequency or TF-IDF weighting.
4. The semantic information from Step 3 is combined with the word embeddings from Step 2 through a fusion strategy (*concat* or *replace*).
5. Classification with a DNN classifier.

The above workflow is illustrated in Figure 7.

4 Experimental evaluation

In this section, we outline the experiments performed to evaluate our semantic augmentation approaches for text classification. In Section 4.1, we describe the datasets and the experimental setup, in Section 4.2, we present and discuss the obtained results, and in Section 4.3, we compare our approach to related studies.

4.1 Datasets and experimental setup

We use the 20-Newsgroups dataset (Lang, 1995)[7], a popular text classification benchmark. This corpus consists of $11,314$ and $7,532$ training and test instances of

---

[7] http://qwone.com/~jason/20Newsgroups/
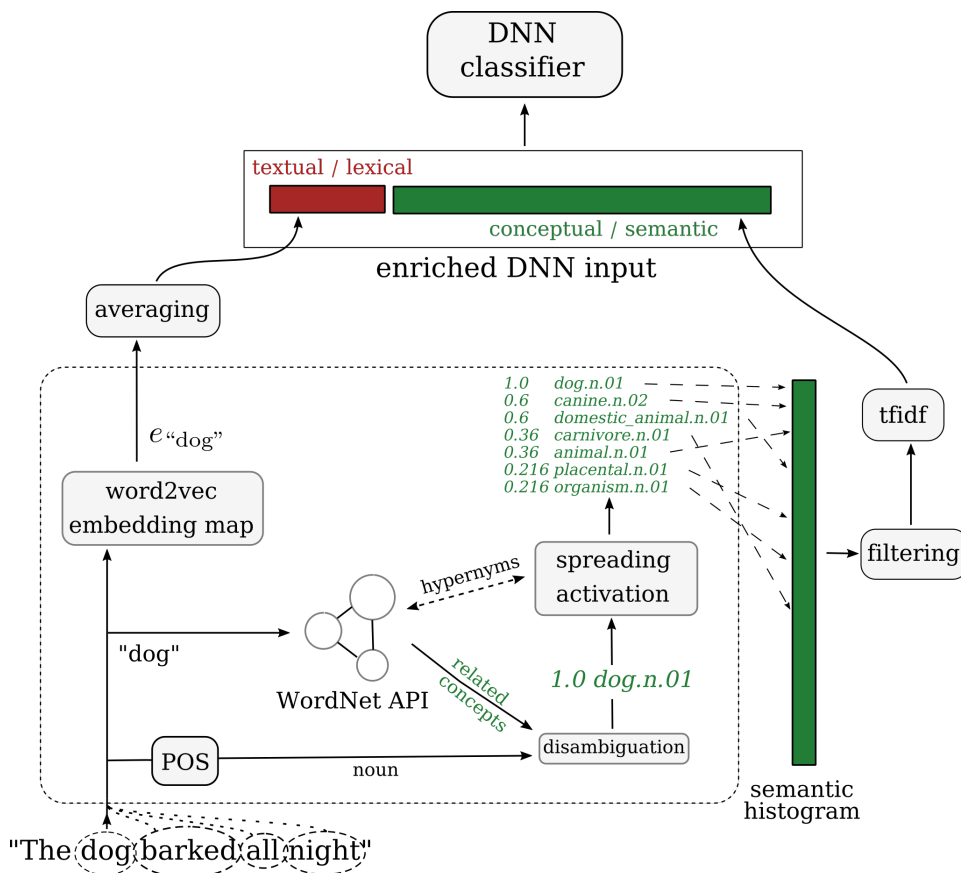
Fig. 7. An example of the semantic augmentation process, leading up to classification with a DNN classifier. The image depicts the case of *concat* fusion, *i.e.*, the concatenation of the word embedding with the semantic vector. The dashed box is repeated for each word in the document. Green / red colors denote semantic / textual information, respectively.

user USENET posts, spanning 20 categories (or "newsgroups") that pertain to different discussion topics (*e.g. alt.atheism*, *sci.space*, *rec.sport.hockey*, *comp.graphics*, *etc.*). The number of instances per class varies from 377 to 600 for the training set, and from 251 to 399 for the test set, while the mean number of words is 191 and 172 per training and test document, respectively. We use the "bydate" version, in which the train and test samples are separated in time (*i.e.*, the train and the test set instances are posted before and after a specific date).

Additionally, we utilize the Reuters-21578[8] dataset, which contains news articles that appeared on the Reuters financial newswire in 1987 and are commonly used for text classification evaluation. Using the traditional "ModApte" variant, the corpus

---

[8] http://www.daviddlewis.com/resources/testcollections/reuters21578/

comprises $9,584$ and $3,744$ training and test documents, respectively, with a labelset of 90 classes. The latter correspond to categories related to financial activities, ranging from consumer products and goods (*e.g. grain, oilseed, palladium*) to more abstract monetary topics (*e.g. money-fx, gnp, interest*). The dataset is extremely imbalanced, ranging from 1 to $2,877$ training instances per class, and from 1 to $1,087$ test instances per class. The mean number of words is approximately 92, for both training and test documents. Most instances are labelled with a single class, with few of them having a multi-label annotation (up to 15 labels per instance). While hierarchical relationships exist between the classes, we do not consider them in our evaluation.

Given that we are only interested in single-label classification, we treat the dataset as a single-labelled corpus by using all sample and label combinations that are available in the dataset. This results into a noisy labelling that is typical among folksonomy-based annotation (Peters and Stock, 2007). In such cases lack of annotator agreement occurs regularly and increases the expected discrimination difficulty of the dataset, as we discard neither superfluous labels nor multi-labelled instances.

The technical details of each dataset are summarized in Table 1. Apart from sample and word count information, we additionally include: (a) quantities pertaining to the part-of-speech information useful for the *POS* disambiguation method; (b) the amount of semantic information minable from the text. The POS annotation count and the synset/concept counts are expressed as ratios with respect to the number of words per document.

We note that no contextual or domain-specific information is employed in our experiments; reflected by the broad task handled in this work (*i.e.* text classification), both datasets are handled in an identical manner by every configuration examined in the experimental evaluation in the following section. This decision introduces an applicability / performance tradeoff: first, the lack of special treatment enables each configuration to be readily applicable to any dataset, with the experimental evaluation better reflecting the generalization capability of the approach. This translates to a single processing pipeline for all datasets, without over-engineered solutions on case-specific details. On the other hand, such a strategy may sacrifice performance gains attainable via accommodating domain or dataset-specific issues. However, we believe that demonstrating the generalization ability of the process is more important and, thus, we focus on this aspect in the experiments.

We used python with Keras[9] (Chollet et al., 2015) and TensorFlow[10] (Abadi et al., 2016) to build the neural models. All experiments are reproducible via the code that is available on GitHub[11]. Document pre-processing was performed with Keras and NLTK[12] (Loper and Bird, 2002). We use WordNet version 3.0 for semantic

---

[9] https://keras.io/
[10] https://www.tensorflow.org/
[11] https://github.com/npit/nlp-semantic-augmentation/tree/jnle
[12] https://www.nltk.org/

	20-Newsgroups		Reuters	
attribute	train	test	train	test
samples	11,314	7,532	9,584	3,744
class samples	377 - 600	251 - 399	1 - 2,877	1 - 1,087
words	191.164 (587.7)	172.196 (471.37)	92.532 (92.03)	92.899 (105.25)
POS	0.716 (0.07)	0.713 (0.06)	0.672 (0.10)	0.669 (0.10)
WordNet	0.572 (0.09)	0.566 (0.09)	1.479 (0.37)	1.381 (0.38)

Table 1. *Technical characteristics of the 20-Newsgroups and Reuters datasets. Class samples refers to the range of the number of instances per class, while the last three rows report mean values, with the corresponding standard deviation in parenthesis. The values in the last two rows (POS, WordNet) are expressed as ratios with respect to the number of words per document.*

information extraction via the interface available in NLTK[13]. The datasets and semantic resources were acquired via the scikit-learn[14] and NLTK APIs[15].

4.2 Results

We now present the results of our experimental evaluation, discussing the performance of each method per dataset.

4.2.1 20-Newsgroups

In the following tables, we present results in accuracy and macro F1-score (columns "accuracy" and "ma-f1", respectively), in terms of mean values over 5 folds. We omit standard deviation scores in favor of compactness and since they consistently fall below 0.005. In the "enrichment" column, the concatenation of the embedding and the semantic vector is denoted by "concat", whereas "replace" indicates the replacement of the former with the latter. The "features" column reports the use of raw concept frequencies ("freq") or TF-IDF weights ("tfidf"). The "disam" column indicates the disambiguation strategy, i.e. "basic", "POS" and "context" corresponding to *basic*, *POS content-embedding* respectively. The "+spread" suffix denotes the use of the spreading activation that is outlined in Section 3.2.2. Finally, the dimensionality of each data vector is reported in the "dim" column. All results are obtained by training and evaluating the DNN model that is described in Section 3.3. Note that we include two baseline methods: the first row corresponds to the *majority classifier*, which always selects the class with the most samples in

---

the training dataset, while the second row corresponds to word2vec embeddings, without any semantic augmentation applied in the vector ("embedding-only").

In Table 2, we present the experimental results for the 20-Newsgroups dataset, which give rise to the following observations:

- The introduction of semantic information brings gains in both accuracy and macro F1-score. The best combination concatenates raw frequency-based concept vectors to the word embeddings, with disambiguation applied according to the *basic* selection strategy.
- Regarding the concept selection method, projecting semantic vectors in the embedding vector space does not improve performance. Part-of-speech filtering yields marginally inferior results than selecting the first retrieved synset from the WordNet API, which is the simplest and best-performing approach.
- The 3-rd order hypernymy propagation via the spreading activation mechanism does not improve the baseline semantic augmentation in a consistent manner.
- Concatenating the word embedding with the semantic vector consistently outperforms the replacement of the former with the latter to a large extent.
- Raw concept frequencies always outperform the TF-IDF normalized weights.

Given these results and observations, we move on to an error analysis of our system by examining the performance of our best-performing configuration in more detail. To this end, Figure 8(a) illustrates the classification error via the confusion matrix for the best-performing configuration. To aid visualization, the diagonal has been removed. We observe that misclassification is approximately fairly concentrated in the first 6 classes[16] (*i.e.*, *alt.atheism*, *comp.graphics*, *comp.os.ms-windows.misc*, *comp.sys.ibm.pc.hardware*, *comp.sys.mac.hardware* and *comp.windows.x*), with peaks appearing at classes 15 and 16 (*soc.religion.christian* and *talk.politics.guns*, respectively). Additionally, Figure 8(b) depicts the label-wise performance for the best-performing configuration. We can see that most labels perform at an F1-score above a value of 0.6, with class 10 (*rec.sport.hockey*) being the easiest to handle by our classifier and class 19 (*talk.religion.misc*) being the most difficult.

Furthermore, Table 3 presents indicative misclassification cases selected from the erroneous prediction of our best-performing configuration. A number of patterns and explanations in these errors are identified by a manual analysis of the results, hereby outlined by selected examples. Specifically, four cases are identified ((a) through (d)). Example instances for each case are referred to by an ID (*e.g.* a1, a2, b1, *etc.*). For each instance we illustrate the true label, the wrong prediction made by our system, and indicative segments found in the instance text.

- First, our system often labels instances with very similar / plausible alternative annotations to the ground truth, which could be however arguably

---

[16] For a complete reference regarding the mapping of numeric indexes to class names, refer to Table 14 in the appendix.

regarded as semantically valid – even by human evaluators – given the instance text. This case is presented in the table group (a) - Semantically similar labels. In example a1, the label *alt.atheism* is applied instead of the true label *talk.religion.misc*, on a 20-Newsgroups instance discussing theism and the conversion of believers to/from atheism, while example a3 is misclassified to *comp.sys.mac.hardware* rather than *sci.electronics*, with the discussion in the text dealing with several computer parts.

- Similarly, our system deviates from the ground truth due to ambiguity and multiple distinct thematic topics in the textual content of many instances. These texts thus approach a multi-label nature with respect to the available classes. This case is reflected in group (b) – Ambiguous / equivocal instances – where we list discussions that involve multiple terms and keywords connected to many classes. For example, b1 contains references to hardware but has a ground truth generally related to the Windows operating system, while b2 mentions multiple graphics file formats but the true label is the X window system. Instances b3 and b4 are mislabelled as sales-related posts from related keywords and terms (*e.g.* "revenue", "business", "for sale", *etc.*), instead of classes linked to specific products and objects of discussion. Finally, text b5 features a lengthy discussion on US abortion legislation which was labelled as a political posting by our system, rather than a religious one.

- Further, there are cases where the content of the text is critically linked to a single or very few mentions of a named entity, that our model either disregards or the available data is not sufficient to leverage (table section (c) - Critical named entity). Such cases are illustrated with examples c1 and c2, where knowledge that "Jack Morris" is a baseball player or that "VAX" is an IBM machine would be required to reach the correct conclusions.

- Additionally, some errors can arise due to the disambiguation method failing to produce to the correct sense, given the context. These cases are presented in group (d) - Context miss. In instance d1, the system associates mentions of battery features to description of product aspects and/or bad reviews, predicting a sales-related text. In example d2, our model gives a very large weight on "The Devil Reincarnate" user handle, deducing a religion class from that very mention and undervaluing the other text terms. In d3, mentions of "disk" are likely linked to the hard disk drive mechanical component, rather than the software-centric sense in the context of operating systems, leading to a corresponding misclassification.

Cases of classification error not included below may be harder to explain; potential causes for them could involve data outliers, classifier bias due to sample / instance size imbalances, *etc.*

Having performed an error analysis of our model, we assess the statistical significance of the performance improvements introduced by each semantic enrichment configuration. To this end, we present in Table 4 the pairwise t-test results of the experimental configurations presented above, with respect to the "embedding-only" baseline method. We can see that all configurations achieve significantly different

enrichment	features	disam	accuracy	ma-f1	dim
N/A	majority-base	N/A	0.005	0.053	N/A
N/A	embedding-only	N/A	0.724	0.716	50
concat	freq	basic	**<u>0.784</u>**	**<u>0.790</u>**	18,157
concat	tfidf	basic	<u>0.752</u>	<u>0.760</u>	18,157
replace	freq	basic	<u>0.763</u>	<u>0.769</u>	18,107
replace	tfidf	basic	0.643	0.648	18,107
concat	freq	pos	<u>0.778</u>	<u>0.784</u>	19,549
concat	tfidf	pos	<u>0.755</u>	<u>0.762</u>	19,549
replace	freq	pos	<u>0.764</u>	<u>0.770</u>	19,499
replace	tfidf	pos	0.648	0.652	19,499
concat	freq	basic+spread	<u>0.775</u>	<u>0.782</u>	22,116
concat	tfidf	basic+spread	*<u>0.755</u>*	*<u>0.762</u>*	22,116
replace	freq	basic+spread	<u>0.761</u>	<u>0.768</u>	22,066
replace	tfidf	basic+spread	*0.652*	0.656	22,066
concat	freq	pos+spread	<u>0.778</u>	<u>0.784</u>	23,546
concat	tfidf	pos+spread	*0.762*	<u>0.755</u>	23,546
replace	freq	pos+spread	*<u>0.768</u>*	*<u>0.762</u>*	23,496
replace	tfidf	pos+spread	*<u>0.766</u>*	*<u>0.760</u>*	23,496
concat	tfidf	context	0.714	<u>0.722</u>	803
concat	freq	context	0.719	<u>0.725</u>	803
replace	tfidf	context	0.563	0.569	753
replace	freq	context	0.673	0.679	753

Table 2. *20-Newsgroups main experimental results.* <u>*Underlined*</u> *values outperform the "embedding-only" baseline method, while* **bold** *values indicate the best dataset– wise performance. Values in italics represent a performance boost achieved by the spreading activation in comparison to the identical configuration without it. "N/A" stands for non-applicable.*

results at a 5% confidence level, with most configurations also performing even more consistently differently, at a 1% confidence level. Regarding the statistical significance of hypernymy propagation, we assess it by comparing its presence and absence, *i.e.*, "X+spread" rows against "X" rows. Min / max p values ranged from 9.97e−06 to 7.50e−01 for the accuracy score and from 1.29e−05 to 9.78e−01 for the macro F1-score (due to lack of space, we omit detailed results). There, statistical significance was achieved at the 5% confidence level for the majority of configurations, with many configurations (including those composed of *POS* concept selection, *replace* fusion and TF-IDF weights) surpassing the 1% confidence level for accuracy.

Next, we examine the effect of dimensionality reduction on the performance of our

id	true / predicted label	indicative text
	(a) - Semantically similar labels	
a1	talk.religion.misc / alt.atheism	scriptures are the whole truth / convert non-theism / religion is a nice fantasy
a2	talk.religion.christian / talk.religion.misc	evangelical counter-cult organization / World of Faith / historic Christian faith
a3	comp.sys.mac.hardware / sci.electronics	Intel processor / HP-IB connector / Mac board NATIONAL INSTRUMENTS
	(b) - Ambiguous / equivocal instances	
b1	comp.os.ms-windows.misc / comp.sys.ibm.pc.hardware	McAFree anti-virus program / scan entire hard disk / my friend's machine
b2	comp.windows.x / comp.graphics	converters for xpm converting GIFs/JPEGs/PS / xpm format
b3	sci.electronics / misc.forsale	lost revenue due to pirates / business environment / purchase a few copies
b4	rec.autos / misc.forsale	MR2 seats for sale / gave the seller / ask questions when buying
b5	talk.religion.misc / talk.politics.misc	Tieing Abortion to Health Reform / Clinton tried to block funding / government to pay
	(c) - Critical named entity	
c1	rec.sport.baseball / rec.autos	this Jack Morris fella / season's only just started / how Morris is doing
c2	comp.sys.ibm.pc.hardware / comp.sys.mac.hardware	VAX/VMS VNEWS / PC Power cooling fan / heat sink grease
	(d) - Context miss	
d1	sci.electronics / misc.forsale	Lead Acid batteries / battery goes dead / plates of the battery
d2	rec.autos / talk.religion.misc	The Devil Reincarnate / commercial cars today / VW Golf/Passat
d3	comp.os.ms-windows.misc / comp.sys.ibm.pc.hardware	disk copy / PC has 4MB RAM / allocate more memory

Table 3. *Misclassification cases for the best-performing configuration over the 20-Newsgroups dataset, where (a) the predicted label is semantically similar to the ground truth, (b) the test instance can be reasonably considered semantically ambiguous, given the labelset, (c) the error is related to the existence of critical named entities, or (d) the error is linked to context misidentification. True / predicted labels refer to the instance ground truth and the erroneous prediction of our model for that test instance, respectively. The listed slash-separated text segments from each instance are indicative samples believed to have had a contribution to misclassification.*

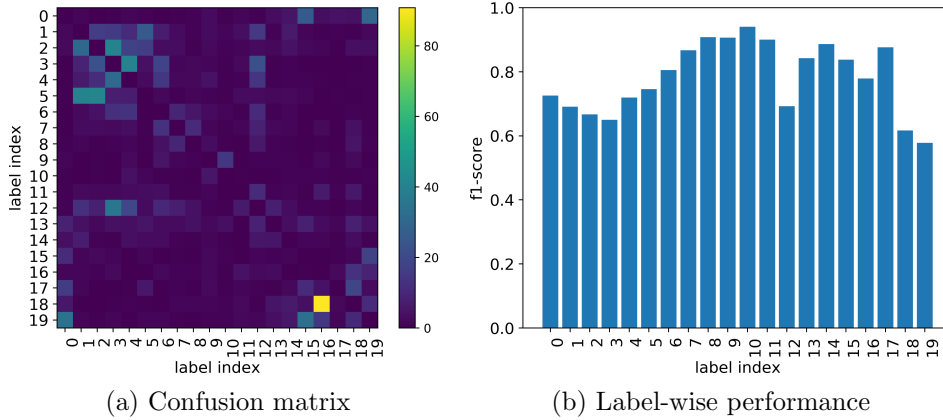(a) Confusion matrix   (b) Label-wise performance

Fig. 8. (a) The diagonal-omitted confusion matrix, and (b) the label-wise performance chart for our best performing configuration over the 20-Newsgroups dataset.

semantic vectors. Table 5 reports the performance when discarding concepts with a raw concept-wise frequency of at least 20 occurrences, while Table 6 corresponds to keeping only the top-50 concepts, in terms of dataset-wise frequency, so as to match the word embedding dimension. We apply both filtering techniques on all configurations of our approach, except for hypernymy propagation and context embedding, due to space limitations. We observe the following patterns:

- The frequency threshold of 20 reduces the dimensionality of the semantic vectors by more than 50%, at a minor cost in classification accuracy. Still, all configurations – excluding TF-IDF with word embedding replacement – surpass the "embedding-only" baseline. In fact, the best configuration of the main experiments (*i.e.*, concatenation fusion with *basic*-selected raw frequencies) maintains a performance very close to its original one.
- When keeping the 50 most-frequent concepts dataset-wise, only concatenation fusion is comparable to the baseline scores. For these cases, TF-IDF weights perform better than the raw concept frequencies, being very close to the baseline results. However, no configuration surpasses the baseline scores.

4.2.2 Reuters

Table 8 reports the experimental results over the Reuters-21578 dataset, which give rise to the following conclusions:

- We observe the same performance patterns as in the case of the 20-Newsgroups dataset. The semantic augmentation outperforms the "embedding-only" baseline, but not in all cases. The feature vectors are again high-dimensional, although considerably shorter than the 20-Newsgroups dataset. This should be attributed to the fewer training documents and the significantly shorter documents (in terms of words) in the Reuters dataset.

enrichment	features	disam	accuracy	ma-f1
concat	freq	basic	5.92e−06**	4.08e−06**
concat	tfidf	basic	2.54e−05**	6.08e−06**
replace	freq	basic	3.21e−04**	8.24e−05**
replace	tfidf	basic	4.99e−07**	3.62e−07**
concat	freq	pos	4.89e−05**	2.13e−05**
concat	tfidf	pos	8.65e−06**	2.68e−06**
replace	freq	pos	3.48e−04**	1.11e−04**
replace	tfidf	pos	6.73e−06**	9.85e−06**
concat	freq	basic+spread	3.23e−07**	7.51e−08**
concat	tfidf	basic+spread	5.44e−05**	1.88e−05**
replace	freq	basic+spread	7.21e−05**	1.85e−05**
replace	tfidf	basic+spread	4.60e−06**	8.47e−06**
concat	freq	pos+spread	1.73e−04**	8.66e−05**
concat	tfidf	pos+spread	1.84e−05**	2.65e−06**
replace	freq	pos+spread	4.40e−04**	7.65e−05**
replace	tfidf	pos+spread	2.17e−04**	5.09e−05**
concat	tfidf	context	2.51e−03**	1.72e−02*
concat	freq	context	2.95e−02*	2.86e−03**
replace	tfidf	context	3.52e−05**	8.85e−07**
replace	freq	context	3.52e−05**	8.21e−05**

Table 4. *20-Newsgroups main experimental pairwise t-test results, with respect to the "embedding-only" baseline. Single and double-starred values represent statistical significance at 5% and 1% confidence levels, respectively.*

enrichment	weights	disam	accuracy	ma-f1	dim
N/A	majority-base	N/A	0.005	0.053	N/A
N/A	embedding-only	N/A	0.724	0.716	50
concat	freq	basic	<u>0.779</u>	<u>0.786</u>	7,092
concat	tfidf	basic	<u>0.754</u>	<u>0.761</u>	7,092
replace	freq	basic	<u>0.764</u>	<u>0.771</u>	7,042
replace	tfidf	basic	0.674	0.679	7,042
concat	freq	pos	<u>0.777</u>	<u>0.784</u>	7,323
concat	tfidf	pos	<u>0.752</u>	<u>0.759</u>	7,323
replace	freq	pos	<u>0.763</u>	<u>0.770</u>	7,273
replace	tfidf	pos	0.673	0.678	7,273

Table 5. *Experiments over the 20-Newsgroups dataset for a concept-wise frequency threshold of 20.* <u>Underlined</u> *values outperform the "embedding-only" baseline.*

enrichment	weights	disam	accuracy	ma-f1	dim
N/A	majority-base	N/A	0.005	0.053	N/A
N/A	embedding-only	N/A	0.724	0.716	50
concat	freq	basic	0.705	0.697	100
concat	tfidf	basic	0.723	0.715	100
replace	freq	basic	0.221	0.212	50
replace	tfidf	basic	0.128	0.081	50
concat	freq	pos	0.705	0.697	100
concat	tfidf	pos	0.722	0.714	100
replace	freq	pos	0.219	0.209	50
replace	tfidf	pos	0.126	0.079	50

Table 6. *Experiments over the 20-Newsgroups dataset for a dataset-wise frequency threshold of* 50. *No configuration outperforms the "embedding-only" baseline.*

- The best performance is obtained when using raw frequency features, concatenated to the word embedding with part-of-speech selection, in combination with hypernymy propagation via spreading activation. This configuration gives the best accuracy (0.749), closely followed by the same configuration with *basic* selection. For the macro F1-score, replacing the word embedding with frequency-based vectors, part-of-speech selection and hypernymy propagation performs the best (0.378). In fact, the replace fusion strategy holds the top 3 configurations with respect to macro F1-score.
- Regarding concept selection, the *basic* and the *POS* techniques result in very similar performance. Context embedding selection exhibits poor accuracy, performing under the "embedding-only" baseline, save for marginal improvements in macro F1-score.
- Spreading activation performs the best here.
- Comparing the replacement fusion strategy with the concatenation one almost always favors the latter, with considerable performance difference.
- The TF-IDF weights always perform under the raw concept frequencies, as in the 20-Newsgroups case.

Similarly to the 20-Newsgroups dataset case, we move on to the error analysis, with Figure 9(a) depicting the confusion matrix with the misclassified instances (*i.e.* diagonal entries are omitted). For better visualization, it illustrates only the 26 classes with at least 20 samples, due to the large number of classes in the Reuters dataset. We observe that the misclassification occurrences are more frequent, but less intense than those in the 20-Newsgroup dataset. Noticeable peaks are in classes *crude*, *grain* and *money-fx*. Additionally, Figure 9(b) depicts the label-wise performance of our best configuration. We observe that it varies significantly, with classes

like *earn* and *acq* achieving excellent performance, while others perform rather poorly, *e.g.*, *soybean* and *rice*.

Moving on to a manual inspection of misclassified instances from the Reuters test set, Table 7 presents such examples, produced when using our best-performing configuration. As in the previous section, we pair the listed examples with the true and predicted labels, indicative terms in the text and possible explanations for the erroneous result, given the instance content:

- Firstly, we observe that Reuters includes labels with considerable semantic similarity to others, presented in the first table group ((a) - Semantically similar labels). For example, the class *coconut* is close to *coconut-oil*, while labels like *grain*, *rye* and *wheat* cover varying levels of specificity among types of plant grains and crops. Instance a1 includes coconut production-related terms, while a2 contains multiple mentions of wheat and its pricing and a3 details production information of a variety of grain crops. Likewise, generic and specific classes for seed crops (*e.g. oilseed*, *rapeseed*) as well as vegetable seed oils (*i.e. rape-oil*, *veg-oil*) can be considered as exhibiting semantic overlap. For example, instance a5 is misclassified as such, with its text containing multiple references to various kinds of vegetable oils. For these examples, mislabelling is manifested from the generic ground truth to a deviation to a more specific class, or vice-versa.
- Secondly, scenarios where polysemous instances are estimated to contribute to misclassification are covered in the error category (b) - Ambiguous / equivocal instances. There, we can find test documents with, *e.g.*, the aluminum ground truth class (*alum*) being mislabelled to *gold* and *yen*, with mentions to the precious metal and the Japanese currency in instances b1 and b2 respectively being a core, non-trivial theme in the text. Similarly, multiple themes can be identified in examples b3 (cocoa and information detailing its national production), b4 (the dollar and foreign monetary exchange, with multiple terms pertaining to the latter) and b5 (shipping and transport of wheat).

Moving on to significance testing, we examine the performance difference between each semantic augmentation configuration and the "embedding-only" baseline, reporting in Table 9 the corresponding pairwise t-test results. We observe that most configurations achieve significantly different performance at a 5% confidence level, while all hypernymy propagation runs perform even more consistently better, at a 1% confidence level. Runs with *context-embedding* disambiguation and *concat* fusion do not perform significantly different than the baseline at the examined confidence levels. Regarding the significance of the hypernymy propagation runs with respect to the semantic runs without it, the improvements introduced by all configurations of the former are significant at a 1% confidence level: the p values range from 3.64e−08 to 1.37e−06 for accuracy, and from 8.54e−08 to 2.57e−06 for macro F1-score (we omit detailed results due to lack of space).

Finally, applying the dimensionality reduction methods to the semantic vectors yields the results reported in Tables 10 and 11, which can be summarized as follows:

- The trade-off introduced by the concept-wise frequency threshold of 20 is

id	true / predicted label	indicative text
	(a) - Semantically similar labels	
a1	coconut-oil / coconut	Philippine Coconut Authority / five-year coconut production cycle / coconut product export
a2	grain / wheat	tonnes of soft wheat / USDA approval for wheat price / Continental Grain Co
a3	rye / grain	Danish crops two weeks behind / barley [...] rapeseed / winter wheat, [...] winter rye
a4	oilseed / rapeseed	4,000 tonnes canadian rapeseed / Canadian rapeseed overnight / at an undisclosed price
a5	rape-oil / veg-oil	rise in edible oil demand / soybean oil [...] rapeseed oil / other origin oils
	(b) - Ambiguous / equivocal instances	
b1	alum / gold	gold, silver, copper and aluminum / Gold futures which previously had a limit of / the metals market
b2	alum / yen	swapping 32 billion yen / capitalisation to 147 billion yen / Japanese shareholders
b3	cocoa / gnp	German grindings expected to [...] / Grindings rose to 55,190 tonnes / compared to [...] European countries
b4	dlr / money-fx	dilemma over monetary policy / further appreciation of the mark / a weaker dollar would be risky
b5	ship / grain	ship prepares to load wheat / urgently needed wheat for Fiji / Australian Wheat Board spokesman

Table 7. *Misclassification cases for the best-performing configuration over the Reuters dataset, where (a) the predicted label is semantically similar to the ground truth, or (b) the test instance can be reasonably considered semantically ambiguous, given the labelset. True / predicted labels refer to the instance ground truth and the erroneous prediction of our model for that test instance, respectively. The listed slash-separated text segments from each instance are indicative samples believed to have had a contribution to misclassification.*

similar to the 20-Newsgroups case: the dimensionality of the semantic vectors is radically reduced, at a negligible cost in classification performance.

- The combination of TF-IDF weights with the replace fusion strategy exhibits the worst performance, underperforming the "embedding-only" baseline, contrary to the rest of the configurations.
- The 50 most-frequent concepts filtering reduces performance even further, with no configuration surpassing the "embedding-only" baseline. Similar to the 20-Newsgroups case, the combination of the concatenation fusion strategy

enrichment	weights	disam	accuracy	ma-f1	dim
N/A	majority-base	N/A	0.2903	0.005	N/A
N/A	embedding-only	N/A	0.725	0.295	50
concat	freq	basic	<u>0.747</u>	<u>0.359</u>	8,534
concat	tfidf	basic	0.723	<u>0.320</u>	8,534
replace	freq	basic	<u>0.744</u>	<u>0.362</u>	8,484
replace	tfidf	basic	0.628	0.222	8,484
concat	freq	pos	<u>0.746</u>	<u>0.349</u>	9,135
concat	tfidf	pos	0.723	<u>0.318</u>	9,135
replace	freq	pos	<u>0.744</u>	<u>0.372</u>	9,085
replace	tfidf	pos	0.627	0.230	9,085
concat	freq	basic+spread	*0.748*	*0.364*	11,358
concat	tfidf	basic+spread	0.720	*0.322*	11,358
replace	freq	basic+spread	*0.745*	*0.376*	11,308
replace	tfidf	basic+spread	*0.643*	*0.232*	11,308
concat	freq	pos+spread	***0.749***	*0.365*	11,966
concat	tfidf	pos+spread	0.720	<u>0.318</u>	11,966
replace	freq	pos+spread	*0.746*	***0.378***	11,916
replace	tfidf	pos+spread	*0.644*	*0.243*	11,916
concat	tfidf	context	<u>0.726</u>	<u>0.301</u>	803
concat	freq	context	0.724	0.291	803
replace	tfidf	context	0.621	0.147	753
replace	freq	context	0.706	0.277	753

Table 8. *Reuters main experimental results.* <u>*Underlined*</u> *values outperform the "embedding-only" baseline, while* **bold** *values indicate the best dataset-wise performance. Values in* italics *denote a performance boost by the spreading activation, with respect to the identical configuration without it.*

with the TF-IDF weights performs the best, matching the macro F1-score of the baseline run.

4.3 Discussion

We now interpret the experimental findings in relation to the research questions posed in Section 1 and compare our approach with the state-of-the-art in the field.

4.3.1 Addressing the research questions

In light of the experimental results, we revisit the research questions stated in the introduction of the paper.

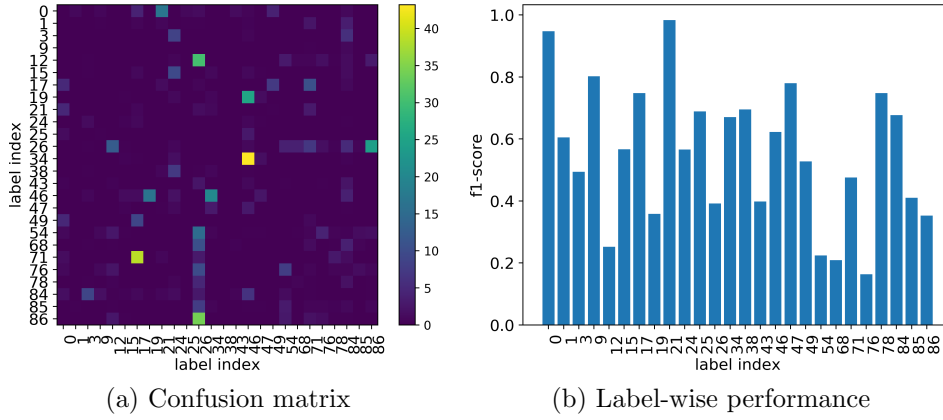(a) Confusion matrix   (b) Label-wise performance

Fig. 9. (a) The diagonal-omitted confusion matrix, and (b) the label-wise performance chart for our best performing configuration over the Reuters dataset. For better visualization, only the 26 classes with at least 20 samples are illustrated.

enrichment	weights	disam	accuracy	ma-f1
concat	freq	basic	1.80e−04**	1.91e−04**
concat	tfidf	basic	1.29e−01	4.33e−03**
replace	freq	basic	1.04e−04**	2.00e−04**
replace	tfidf	basic	1.32e−06**	7.21e−05**
concat	freq	pos	3.11e−05**	2.40e−03**
concat	tfidf	pos	3.72e−01	1.09e−02*
replace	freq	pos	2.06e−05**	1.17e−04**
replace	tfidf	pos	2.93e−07**	8.84e−04**
concat	freq	basic+spread	2.30e−06**	6.25e−10**
concat	tfidf	basic+spread	3.52e−07**	5.78e−10**
replace	freq	basic+spread	4.41e−07**	5.10e−10**
replace	tfidf	basic+spread	6.94e−08**	1.03e−09**
concat	freq	pos+spread	1.94e−07**	1.83e−11**
concat	tfidf	pos+spread	4.29e−07**	4.21e−10**
replace	freq	pos+spread	1.00e−06**	2.66e−11**
replace	tfidf	pos+spread	1.61e−07**	9.09e−10**
concat	tfidf	context	4.16e−01	2.74e−01
concat	freq	context	2.56e−01	5.03e−01
replace	tfidf	context	1.99e−07**	2.51e−07**
replace	freq	context	3.49e−05**	3.00e−03**

Table 9. *Reuters main experimental pairwise t-test results, with respect to the "embedding-only" baseline. Single and double-starred values represent statistical significance at 5% and 1% confidence levels, respectively.*

enrichment	weights	disam	accuracy	ma-f1	dim
N/A	majority-base	N/A	0.2903	0.005	N/A
N/A	embedding-only	N/A	0.725	0.295	50
concat	freq	basic	<u>0.747</u>	<u>0.358</u>	3,286
concat	tfidf	basic	<u>0.726</u>	<u>0.309</u>	3,286
replace	freq	basic	<u>0.745</u>	<u>0.356</u>	3,236
replace	tfidf	basic	0.651	0.231	3,236
concat	freq	pos	<u>0.748</u>	<u>0.363</u>	3,308
concat	tfidf	pos	<u>0.726</u>	<u>0.319</u>	3,308
replace	freq	pos	<u>0.745</u>	<u>0.362</u>	3,258
replace	tfidf	pos	0.653	0.241	3,258

Table 10. *Experiments over the Reuters dataset for a concept-wise frequency threshold of* 20. <u>*Underlined*</u> *values outperform the "embedding-only" baseline.*

enrichment	weights	disam	accuracy	ma-f1	dim
N/A	majority-base	N/A	0.2903	0.005	N/A
N/A	embedding-only	N/A	0.725	0.295	50
concat	freq	basic	0.718	0.277	100
concat	tfidf	basic	0.724	0.293	100
replace	freq	basic	0.635	0.155	50
replace	tfidf	basic	0.577	0.045	50
concat	freq	pos	0.717	0.275	100
concat	tfidf	pos	0.725	0.290	100
replace	freq	pos	0.637	0.154	50
replace	tfidf	pos	0.581	0.048	50

Table 11. *Experiments over the Reuters dataset for a dataset-wise frequency threshold of* 50. *No configurations outperforms the "embedding-only" baseline.*

1. **Can semantic information increase the task performance, when applied in this setting? If so, how much?** Experimental results show that a considerable performance boost is achieved by injecting semantic information in the network input, with the improvement achieving significance at a 5% confidence level. Our semantic augmentation approach, which inserts WordNet concept statistics into the network input, achieves the best average performance when using raw concept frequencies, selecting the first retrieved

concept per word (*basic* strategy) and concatenating the resulting vector to the word2vec embedding.

2. **Do different semantic disambiguation methods affect the above performance increase and how?** Out of the three examined methods (*i.e.*, *basic*, *POS* and *context-embedding*), the first one seems to suffice, the second one appears to have a non-noteworthy effect on the final performance, while the third one performs under the other two across all datasets. We note however, that the performance of *context-embedding* disambiguation depends heavily on the availability of concept-wise context in the semantic resource. Given that WordNet has limited examples per concept (*e.g.*, 70% of the concepts convey a single example sentence), more credible findings should be derived from a wider investigation that includes semantic resources with a richer lexical content. Alternatively, we could relax the context acquisition constraint, *i.e.*, the minimum word count threshold in Section 3.2.2.

3. **What is the effect of enriching the representation with the n-th order hypernymy relations (*e.g.*, through a spreading activation process)?** Initial findings indicate that this effect varies across datasets. For 20-Newsgroups, the hypernymy propagation introduces minor and inconsistent (but statistically significant, at a 5% level) performance boosts, when compared to not using the propagation mechanism. For Reuters, though, the hypernymy propagation gives the best results among the examined configurations, with even greater statistical significance (beyond a 1% confidence level). This discrepancy is probably related to the size of each dataset and to the number of concepts that are extracted from WordNet (the average number of words per document in 20-Newsgroups is double than that in Reuters). Spreading activation apparently does little performance-wise for semantic vectors that are over-saturated, due to the large number of activations that are triggered by the plethora of document words. In fact, spreading activation can be detrimental to performance, if it introduces additional noise by populating the concept histogram with many generic concepts. As a result, further investigation is required in order to draw more solid conclusions regarding the application of this step. Ideally, that investigation should include semantic filtering steps that are applied after hypernymy propagation.

Next, we summarize the main findings obtained from our experimental analysis:

- **Semantic vector dimensionality**: The semantic vectors of our approach are high-dimensional, a common byproduct of frequency-based features. For 20-Newsgroups, applying dimensionality reduction with a static concept-wise frequency threshold set to 20, allows for restricting the vector length by 61%, while retaining 99.36% of the top accuracy score (Table 5). For Reuters, the optimal performance (excluding the hypernymy propagation) stays the same at a 61.5% size reduction (Table 10). On the other hand, dimensionality reduction with the 50 most frequent concepts is evidently too extreme, and the resulting fixed-length vectors cannot be used as substitutes to equal-length word embeddings. This is in line with the next observation, which states that

replacing the embedding vector entirely, rather than augmenting it, consistently deteriorates the classification performance, at least for the datasets and semantic resources we consider. Notably, in this dimensionality reduction scenario – and contrary to the results of both the main and concept-wise thresholded experiments – TF-IDF weights outperform the raw frequency vectors. This probably suggests that TF-IDF is important for semantic frequency-based features, but only at very extreme dimensionality reductions. These findings show that the semantic vector-space features like the ones employed by our approach can be rather noisy, despite representing high-level information. Even though frequency-based concept-wise thresholding works well in our evaluation, both truncation schemes warrant further investigation (*e.g.*, in the context of hypernymy activation propagation and context embedding configurations). It is also interesting to explore the effect of more sophisticated methods for dimensionality reduction on the enriched data produced by our pipeline.

- **Replacement versus concatenation**: The experimental results over both datasets indicate a considerable performance difference between replacing the word embedding with the semantic vector and concatenating the two. The latter consistently outperforms the former, demonstrating that WordNet-based concept information is less important than the distributional and statistical properties that are captured by the word embeddings. This finding carries over to macro-averaged F1-score results for the 20-Newsgroups dataset, but does not apply to the Reuters dataset, where replacing the word embedding yields the best macro-averaged score. This is an interesting divergence from the behaviour in 20-Newsgroups that is possibly explained by the very large class imbalance of Reuters. This means that the semantic concept-based features probably provide useful information for under-represented cases that are inadequately covered by the embedding training. In other words, the concept-based information, which can be readily extracted from a semantic resource, is more suitable for representing minorities (*i.e.*, under-sampled classes), counterbalancing the lack of a critical mass of data that is required by embeddings. This hypothesis is reinforced by the comparison of our approach with vectors pre-trained on very large corpora, as discussed below (Section 4.3).

- **Using raw concept frequencies versus TF-IDF scores**: The experimental results over the 20-Newsgroups dataset contradict previous knowledge about the relationship of token-based weighting schemes, when applied on lexical terms (*e.g.* words / sentences / documents). There, discounting common terms consistently boosts the performance on the task at hand. Yet, the inverse relationship manifests itself in the world of concepts, with TF-IDF weights always underperforming the raw frequency vectors in terms of accuracy. It would be interesting to examine information gain measures on such concept-based features as an alternative method towards identifying the most suitable concepts for augmenting classification performance.

4.3.2 Comparison to the state-of-the-art

We now compare our approach against the state-of-the-art in the field, including both fitted and pre-trained embedding approaches; in all cases, the resulting vectors are used in the classification pipeline in the same way as all other configurations examined and described in this work. Table 12 reports the performance of the majority classifier, the "embedding-only" baseline method which uses only the fitted word2vec embedding, (i) the FastText (Joulin et al., 2017) embeddings, which are fitted to each dataset (ii) the publicly available pre-trained versions of word2vec, FastText and GloVe (Pennington et al., 2014) embeddings, with a dimensionality of 300, 50 and 50, respectively; (iii) the 50-dimensional single-sense vectors used in (Huang et al., 2012) for multi-sense vector generation; (iv) the main sense embedding approaches, *i.e.*, the cluster-based, multi-sense 50-dimensional word vectors from (Huang et al., 2012), the 400-dimensional SensEmbed vectors (Iacobacci et al., 2015) and supersense embeddings (Flekova and Gurevych, 2016). For (iv), we use the publicly available pre-trained vectors of each method, while for (i), (ii) and (iii) we also apply retrofitting (Faruqui et al., 2015) using the same WordNet relations as in our semantic augmentation process. This process runs as a post-processing step for 10 iterations - we experimented with more iterations (up to 50), but observed no improvement.

We can see that our approach outperforms all other state-of-the-art word embeddings – not only the FastText and word2vec vectors that are fitted on the same dataset as our approach, but also the pre-trained versions of FastText, GloVe and word2vec embeddings, which have been trained on a far larger corpora. Only the 300-dimensional pre-trained word2vec surpasses the "embedding-only" baseline. Surprisingly enough, retrofitting the embeddings consistently results in inferior performance, both for the pre-trained ones and for those fitted from scratch. Regarding sense embeddings, both supersenses and SensEmbed vectors work well, surpassing the "embedding-only" baseline, but they do not outperform our approach. The multi-context cluster-based approach underperforms all other configurations.

Further, we include an additional evaluation on two more datasets.

1. The BBC corpus (Greene and Cunningham, 2006) contains articles from the news domain, annotating them with 5 categories (business, politics, entertainment, tech and sports). It consists of 2,225 samples and is evaluated via 10-fold cross validation. Mean document size is approximately 220.1 words.
2. The Ohsumed dataset (Lang, 1995) contains medical texts, specifically 10,433 train and 12,733 test samples, with an approximate mean document size of 108.1 words.

These corpora extend our evaluation to datasets with few classes and small number of samples (as is the case with BBC) and to datasets from a radically different domain (*i.e.*, the medical content of Ohsumed). Table 13 presents the experimental results over the additional datasets for the two main baselines, our best performing configuration as well as top performers from Table 12.

We observe that our approach performs the best for both the additional datasets,

config	Reuters		20-Newsgroups	
system	accuracy	ma-f1	accuracy	ma-f1
majority baseline	0.290	0.005	0.005	0.053
embedding-only	0.725	0.295	0.724	0.716
our approach	**0.749**	**0.378**	**0.784**	**0.790**
other trained embeddings				
FastText (Joulin et al., 2017)	0.732	0.319	0.751	0.743
FastText + retrofitting	0.717	0.260	0.748	0.740
word2vec + retrofitting	0.709	0.248	0.717	0.710
pre-trained embeddings				
glove (Pennington et al., 2014)	0.702	0.275	0.620	0.610
glove + retrofitting	0.684	0.235	0.587	0.575
FastText	0.733	0.310	0.734	0.727
FastText + retrofitting	0.705	0.239	0.706	0.695
word2vec (300-dim)	0.737	0.311	0.721	0.712
word2vec (300-dim) + retrofitting	0.689	0.239	0.476	0.465
single-context (Huang et al., 2012)	0.661	0.227	0.541	0.531
single-context + retrofitting	0.629	0.175	0.464	0.454
pre-trained sense embeddings				
multi-context (Huang et al., 2012)	0.570	0.121	0.430	0.412
SensEmbed (Iacobacci et al., 2015)	0.728	0.308	0.722	0.714
Supersenses (Flekova and Gurevych, 2016)	0.729	0.313	0.733	0.725

Table 12. *Dataset-wise comparison with the state-of-the-art in terms of accuracy and macro-F1 score. <u>Underlined</u> values outperform the "embedding-only" (50-dimensional fitted word2vec) baseline, while **bold** values denote column-wise maxima.*

with the difference being less noticeable on the BBC dataset. The lexical-only word2vec pre-trained embeddings outperform both sense-based approaches, out of which SensEmbed achieves the highest accuracy. Retrofitting word2vec vectors improves the classification results to a minor extent over the Ohsumed dataset.

Regarding experimental results that are reported in relevant studies in the literature, a WordNet-based enrichment approach in (Elberrichi et al., 2008) achieves macro F1-scores of 0.719 and 0.717 for Reuters and 20-Newsgroups, respectively. However, it considers only the 10 most populous classes of the Reuters dataset, unlike our evaluation, which uses all available classes (despite the respective class

	bbc		ohsumed	
system	accuracy	ma-f1	accuracy	ma-f1
majority	0.230	0.075	0.172	0.013
embedding-only	0.970	0.970	0.384	0.300
ours	**0.976**	**0.976**	**0.435**	**0.373**
other pre-trained embeddings				
word2vec	<u>0.973</u>	<u>0.973</u>	0.307	0.244
word2vec + retrofitting	0.880	0.878	0.313	0.250
SensEmbed	0.969	0.969	0.328	0.215
Supersenses	0.852	0.851	0.229	0.148

Table 13. *Evaluation of representative configurations on additional datasets. <u>Underlined</u> values outperform the "embedding-only" baseline, while **bold** values denote column-wise maxima.*

imbalance and folksonomy-based noise explained in Section 4.1). In (Nezreg et al., 2014), the authors apply a frequency-based semantic enrichment approach to heavily reduced training and test portions of the Reuters and 20-Newsgroups datasets, achieving 0.744 and 0.557 precision, respectively (they do not specify the aggregation type). The corresponding scores for our best approach for the Reuters dataset is 0.748 / 0.42 micro and macro average respectively, and a 0.8 micro-macro precision value for 20-Newsgroups. In (Card et al., 2018), the authors examine generative DNN topic models that are able to capture and model semantic information in the form of document meta-data. Applying their approach to document classification, they achieve an optimal accuracy of 0.71 over 20-Newsgroups, which is considerably lower than our top performance (0.79). In (Pilehvar et al., 2017), the authors employ a combination of CNN and LSTM networks with a semantic augmentation process via sense and supersense embeddings extracted from WordNet or Wikipedia via BabelNet. Their best configuration (using WordNet supersense trained embeddings) achieves an F1-score of 0.858 on a subset of 6 classes from 20-Newsgroups, 0.22 on Ohsumed and 0.934 on the BBC dataset; our corresponding scores are 0.790 (on the entire 20-Newsgroups labelset of 20 classes), 0.373 and 0.976. To our knowledge, our approach is directly outperformed only by (Jiang et al., 2018), where the authors use a DBN architecture with softmax classification, achieving an accuracy of 0.8688 and 0.8263 over Reuters and 20-Newsgroups, respectively. Their method does not use semantic information, but employs a larger and deeper neural model than ours together with embedding fine-tuning. We also note that they use a heavily reduced version of the Reuters dataset that consists of just 10 classes.

5 Acknowledgements

6 Conclusion

In this paper, we examined semantic augmentation methods for text classification. To boost classification performance, we extract frequency-based semantic information from the WordNet semantic graph and fuse it with deep neural embeddings. A large experimental evaluation demonstrates the following findings:

1. The use of semantic information from resources such as WordNet significantly improves classification performance, when applied to the input space of the neural model. This straightforward approach outperforms more complex ones in the literature that use the semantic graph to augment the training of embeddings, or to post-process the embedding vectors via retrofitting.
2. Concatenating the embedding and semantic vectors works best, on average, with maximal gains coming at the cost of an increased dimensionality. This increases the computational complexity of learning the classification model. However, this can be radically alleviated by filtering the component concepts. This is best realized by a threshold on minimum concept frequency.
3. Using raw, unnormalized concept frequency scores works best, while the *basic* disambiguation strategy is sufficient to achieve the best performance (with *POS* performing comparably). This seems to further simplify the semantic augmentation process, but a deeper investigation is required on additional and diverse datasets.
4. Hypernymy n-th order propagation via a spreading activation mechanism reinforces the already superior semantic augmentation process to a statistically significant extent.

In the future, we plan to further investigate issues raised from the findings of our experimental analysis, such as the behavior of TF-IDF weighting in frequency-based semantic vectors. We also intend to examine efficient dimensionality reduction approaches for this kind of vector space data, ranging from pooling aggregations (*e.g.* averaging, multiplication, *etc.*) to transformation methods such as PCA (Jolliffe, 2011), LDA (Fukunaga, 2013) or a neural embedding process, similar to other works (Li et al., 2017). We will also test the *context-embedding* approach on additional semantic resources, especially ones that provide a larger supply of example sentences per concept. Finally, we will explore semantically-augmented classification in conjunction with sequence-based neural classifiers, such as Long Short-Term Memory (Hochreiter and Schmidhuber, 1997) and Gated Recurrent Unit (Chung et al., 2014) models, as well in cases where embeddings of a larger dimensionality are employed.

References

Abadi, M., **Barham, P.**, **Chen, J.**, **Chen, Z.**, **Davis, A.**, **Dean, J.**, **Devin, M.**, **Ghemawat, S.**, **Irving, G.**, **Isard, M.**, **and others** 2016. Tensorflow: a system for large-scale machine learning. In *Proceedings of the 12th USENIX conference on Operating Systems Design and Implementation*, pp. 265–283. USENIX Association.

Bengio, Y., **Ducharme, R.**, **Vincent, P.**, **and Jauvin, C.** 2003. A neural probabilistic language model. *Journal of machine learning research*, 3(Feb):1137–1155.

Bengio, Y. and Others 2009. Learning deep architectures for ai. *Foundations and Trends® in Machine Learning*, 2(1):1–127.

Bian, J., **Gao, B.**, **and Liu, T.-Y.** 2014. Knowledge-powered deep learning for word embedding. In *Joint European conference on machine learning and knowledge discovery in databases*, pp. 132–148. Springer.

Bovi, C. D., **Anke, L. E.**, **and Navigli, R.** 2015. Knowledge base unification via sense embeddings and disambiguation. In *Proceedings of the 2015 Conference on Empirical Methods in Natural Language Processing*, pp. 726–736. The Association for Computational Linguistics.

Boyd, S., **Parikh, N.**, **Chu, E.**, **Peleato, B.**, **Eckstein, J.**, **and others** 2011. Distributed optimization and statistical learning via the alternating direction method of multipliers. *Foundations and Trends® in Machine learning*, 3(1):1–122.

Card, D., **Tan, C.**, **and Smith, N. A.** 2018. Neural models for documents with metadata. In *Proceedings of the 56th Annual Meeting of the Association for Computational Linguistics (Volume 1: Long Papers)*, pp. 2031–2040. The Association for Computational Linguistics.

Chen, X., **Liu, Z.**, **and Sun, M.** 2014. A unified model for word sense representation and disambiguation. In *Proceedings of the 2014 Conference on Empirical Methods in Natural Language Processing (EMNLP)*, pp. 1025–1035. The Association for Computational Linguistics.

Chollet, F. and others 2015. Keras. `https://keras.io`.

Chung, J., **Gulcehre, C.**, **Cho, K.**, **and Bengio, Y.** 2014. Empirical evaluation of gated recurrent neural networks on sequence modeling. In *NIPS Deep Learning Workshop*. MIT Press.

Collins, A. M. and Loftus, E. F. 1975. A spreading-activation theory of semantic processing. *Psychological review*, 82(6):407.

Collobert, R. and Weston, J. 2008. A unified architecture for natural language processing: Deep neural networks with multitask learning. In *Proceedings of the 25th international conference on Machine learning*, pp. 160–167. ACM.

Deerwester, S., **Dumais, S. T.**, **Furnas, G. W.**, **Landauer, T. K.**, **and Harshman, R.** 1990. Indexing by latent semantic analysis. *Journal of the American society for information science*, 41(6):391–407.

Elberrichi, Z., **Rahmoun, A.**, **and Bentaalah, M. A.** 2008. Using wordnet for text categorization. *International Arab Journal of Information Technology (IAJIT)*, 5(1).

Faruqui, M., **Dodge, J.**, **Jauhar, S. K.**, **Dyer, C.**, **Hovy, E.**, **and Smith, N. A.** 2015. Retrofitting word vectors to semantic lexicons. In *Proceedings of the 2015 Conference of the North American Chapter of the Association for Computational Linguistics: Human Language Technologies*, pp. 1606–1615. The Association for Computational Linguistics.

Flekova, L. and Gurevych, I. 2016. Supersense embeddings: A unified model for supersense interpretation, prediction, and utilization. In *Proceedings of the 54th Annual Meeting of the Association for Computational Linguistics (Volume 1: Long Papers)*, pp. 2029–2041. The Association for Computational Linguistics.

Fried, D. and Duh, K. 2014. Incorporating both distributional and relational semantics in word representations. *CoRR*, abs/1412.5836.

Fukunaga, K. 2013. *Introduction to statistical pattern recognition*. Elsevier.

Ganitkevitch, J., **Van Durme, B.**, and **Callison-Burch, C.** 2013. Ppdb: The paraphrase database. In *Proceedings of the 2013 Conference of the North American Chapter of the Association for Computational Linguistics: Human Language Technologies*, pp. 758–764. The Association for Computational Linguistics.

Glavaš, G. and Vulić, I. 2018. Explicit retrofitting of distributional word vectors. In *Proceedings of the 56th Annual Meeting of the Association for Computational Linguistics (Volume 1: Long Papers)*, pp. 34–45. The Association for Computational Linguistics.

Goikoetxea, J., **Agirre, E.**, and **Soroa, A.** 2016. Single or multiple? combining word representations independently learned from text and wordnet. In *Proceedings of the Thirtieth AAAI Conference on Artificial Intelligence*, pp. 2608–2614. AAAI Press.

Greene, D. and Cunningham, P. 2006. Practical solutions to the problem of diagonal dominance in kernel document clustering. In *Proceedings of the 23rd international conference on Machine learning*, pp. 377–384. ACM.

Hingmire, S., **Chougule, S.**, **Palshikar, G. K.**, and **Chakraborti, S.** 2013. Document classification by topic labeling. In *Proceedings of the 36th international ACM SIGIR conference on Research and development in information retrieval*, pp. 877–880. ACM.

Hinton, G. E., **McClelland, J. L.**, **Rumelhart, D. E.**, and **others** 1984. *MIT Press*. Carnegie-Mellon University Pittsburgh, PA.

Hochreiter, S. and Schmidhuber, J. 1997. Long short-term memory. *Neural computation*, 9(8):1735–1780.

Huang, E. H., **Socher, R.**, **Manning, C. D.**, and **Ng, A. Y.** 2012. Improving word representations via global context and multiple word prototypes. In *Proceedings of the 50th Annual Meeting of the Association for Computational Linguistics: Long Papers-Volume 1*, pp. 873–882. The Association for Computational Linguistics.

Hung, C. and Wermter, S. 2004. Neural network based document clustering using wordnet ontologies. *International Journal of Hybrid Intelligent Systems*, 1(3-4):127–142.

Iacobacci, I., **Pilehvar, M. T.**, and **Navigli, R.** 2015. Sensembed: Learning sense embeddings for word and relational similarity. In *Proceedings of the 53rd Annual Meeting of the Association for Computational Linguistics and the 7th International Joint Conference on Natural Language Processing (Volume 1: Long Papers)*, pp. 95–105. The Association for Computer Linguistics.

Jiang, M., **Liang, Y.**, **Feng, X.**, **Fan, X.**, **Pei, Z.**, **Xue, Y.**, and **Guan, R.** 2018. Text classification based on deep belief network and softmax regression. *Neural Computing and Applications*, 29(1):61–70.

Joachims, T. 1998. Text categorization with support vector machines: Learning with many relevant features. In *European conference on machine learning*, pp. 137–142. Springer.

Jolliffe, I. 2011. Principal component analysis. In *International encyclopedia of statistical science*, pp. 1094–1096. Springer.

Joulin, A., **Grave, E.**, and **Mikolov, P. B. T.** 2017. Bag of tricks for efficient text classification. *EACL 2017*, 427.

Jurgens, D. A., **Turney, P. D.**, **Mohammad, S. M.**, and **Holyoak, K. J.** 2012. Semeval-2012 task 2: Measuring degrees of relational 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*, pp. 356–364. The Association for Computational Linguistics.

Katz, S. 1987. Estimation of probabilities from sparse data for the language model component of a speech recognizer. *IEEE transactions on acoustics, speech, and signal processing*, 35(3):400–401.

Lang, K. 1995. Newsweeder: Learning to filter netnews. In *Proceedings of the 12th International Conference on Machine Learning*, pp. 331–339. Morgan Kaufmann publishers Inc.

Li, Y., **Wei, B.**, **Yao, L.**, **Chen, H.**, **and Li, Z.** 2017. Knowledge-based document embedding for cross-domain text classification. In *International Joint Conference on Neural Networks (IJCNN)*, pp. 1395–1402. IEEE.

Liu, Y., **Scheuermann, P.**, **Li, X.**, **and Zhu, X.** 2007. Using wordnet to disambiguate word senses for text classification. In *International Conference on Computational Science*, pp. 781–789. Springer.

Loper, E. and Bird, S. 2002. Nltk: the natural language toolkit. In *Proceedings of the ACL-02 Workshop on Effective tools and methodologies for teaching natural language processing and computational linguistics-Volume 1*, pp. 63–70. The Association for Computational Linguistics.

Luong, T., **Socher, R.**, **and Manning, C.** 2013. Better word representations with recursive neural networks for morphology. In *Proceedings of the 17th Conference on Computational Natural Language Learning (CoNLL)*, pp. 104–113. The Association for Computational Linguistics.

Mansuy, T. N. and Hilderman, R. J. 2006. Evaluating wordnet features in text classification models. In *FLAIRS Conference*, pp. 568–573. AAAI Press.

Martin, J. H. and Jurafsky, D. 2009. *Speech and language processing: An introduction to natural language processing, computational linguistics, and speech recognition.* Prentice Hall.

Mcauliffe, J. D. and Blei, D. M. 2008. Supervised topic models. In *Advances in neural information processing systems*, pp. 121–128. Curran Associates, Inc.

Mikolov, T., **Chen, K.**, **Corrado, G.**, **and Dean, J.** 2013a. Efficient estimation of word representations in vector space. In *Proceedings of the First International Conference on Learning Representations*.

Mikolov, T., **Karafiát, M.**, **Burget, L.**, **Cernocký, J.**, **and Khudanpur, S.** 2010. Recurrent neural network based language model. In *INTERSPEECH*, pp. 1045–1048. ISCA.

Mikolov, T., **Kombrink, S.**, **Burget, L.**, **Černockỳ, J.**, **and Khudanpur, S.** 2011. Extensions of recurrent neural network language model. In *International Conference on Acoustics, Speech and Signal Processing (ICASSP)*, pp. 5528–5531. IEEE.

Mikolov, T., **Sutskever, I.**, **Chen, K.**, **Corrado, G. S.**, **and Dean, J.** 2013b. Distributed representations of words and phrases and their compositionality. In *Advances in neural information processing systems*, pp. 3111–3119. Curran Associates, Inc.

Miller, G. A. 1995. Wordnet: a lexical database for english. *Communications of the ACM*, 38(11):39–41.

Miller, G. A., **Leacock, C.**, **Tengi, R.**, **and Bunker, R. T.** 1993. A semantic concordance. In *Proceedings of the workshop on Human Language Technology*, pp. 303–308. The Association for Computational Linguistics.

Morin, F. and Bengio, Y. 2005. Hierarchical probabilistic neural network language model. In *Aistats*, volume 5, pp. 246–252. Society for Artificial Intelligence and Statistics.

Moro, A., **Raganato, A.**, **and Navigli, R.** 2014. Entity linking meets word sense disambiguation: a unified approach. *Transactions of the Association for Computational Linguistics*, 2:231–244.

Navigli, R. 2009. Word sense disambiguation: A survey. *ACM computing surveys (CSUR)*, 41(2):10.

Navigli, R., **Litkowski, K. C.**, **and Hargraves, O.** 2007. Semeval-2007 task 07: Coarse-grained english all-words task. In *Proceedings of the Fourth International Workshop on Semantic Evaluations (SemEval-2007)*, pp. 30–35. The Association for Computational Linguistics.

Navigli, R. and Ponzetto, S. P. 2012. Babelnet: The automatic construction, evaluation and application of a wide-coverage multilingual semantic network. *Artificial Intelligence*, 193:217–250.

Nezreg, H., **Lehbab, H.**, and **Belbachir, H.** 2014. Conceptual representation using wordnet for text categorization. *International Journal of Computer and Communication Engineering*, 3(1):27.

Parker, R., **Graff, D.**, **Kong, J.**, **Chen, K.**, and **Maeda, K.** 2011. English gigaword. *Linguistic Data Consortium*.

Pennington, J., **Socher, R.**, and **Manning, C.** 2014. Glove: Global vectors for word representation. In *Proceedings of the 2014 conference on empirical methods in natural language processing (EMNLP)*, pp. 1532–1543. The Association for Computational Linguistics.

Peters, I. and Stock, W. G. 2007. Folksonomy and information retrieval. *Proceedings of the American Society for Information Science and Technology*, 44(1):1–28.

Pilehvar, M. T., **Camacho-Collados, J.**, **Navigli, R.**, and **Collier, N.** 2017. Towards a seamless integration of word senses into downstream nlp applications. In *Proceedings of the 55th Annual Meeting of the Association for Computational Linguistics (Volume 1: Long Papers)*, pp. 1857–1869. The Association for Computational Linguistics.

Porter, M. 2006. An algorithm for suffix stripping. *Program: electronic library & information systems*, 40(3):211–218.

Rothe, S. and Schütze, H. 2015. Autoextend: extending word embeddings to embeddings for synsets and lexemes. In *Proceedings of the 53rd Annual Meeting of the Association for Computational Linguistics and the 7th International Joint Conference on Natural Language Processing (Volume 1: Long Papers)*, pp. 1793–1803. The Association for Computer Linguistics.

Ruder, S., **Vulić, I.**, and **Søgaard, A.** 2019. A survey of cross-lingual word embedding models. *Journal of Artificial Intelligence Research*, 65:569–631.

Salton, G. and Buckley, C. 1988. Term-weighting approaches in automatic text retrieval. *Information processing & management*, 24(5):513–523.

Scott, S. and Matwin, S. 1998. Text classification using wordnet hypernyms. *Usage of WordNet in Natural Language Processing Systems*.

Scozzafava, F., **Raganato, A.**, **Moro, A.**, and **Navigli, R.** 2015. Automatic identification and disambiguation of concepts and named entities in the multilingual wikipedia. In *Congress of the Italian Association for Artificial Intelligence*, pp. 357–366. Springer.

Vulic, I. and Mrkšic, N. 2018. Specialising word vectors for lexical entailment. In *Proceedings of NAACL-HLT*, pp. 1134–1145. The Association for Computational Linguistics.

Xu, C., **Bai, Y.**, **Bian, J.**, **Gao, B.**, **Wang, G.**, **Liu, X.**, and **Liu, T.-Y.** 2014. Rc-net: A general framework for incorporating knowledge into word representations. In *Proceedings of the 23rd ACM international conference on conference on information and knowledge management*, pp. 1219–1228. ACM.

Yu, M. and Dredze, M. 2014. Improving lexical embeddings with semantic knowledge. In *Proceedings of the 52nd Annual Meeting of the Association for Computational Linguistics (Volume 2: Short Papers)*, pp. 545–550. The Association for Computer Linguistics.

A Dataset label names

In Table 14, we list the mapping between label index and label name, for each dataset examined in the main experimental evaluation.

20-Newgroups					
index	name	index	name	index	name
0	alt.atheism	7	rec.autos	14	sci.space
1	comp.graphics	8	rec.motorcycles	15	soc.religion.christian
2	comp.os.ms-windows.misc	9	rec.sport.baseball	16	talk.politics.guns
3	comp.sys.ibm.pc.hardware	10	rec.sport.hockey	17	talk.politics.mideast
4	comp.sys.mac.hardware	11	sci.crypt	18	talk.politics.misc
5	comp.windows.x	12	sci.electronics	19	talk.religion.misc
6	misc.forsale	13	sci.med		

Reuters					
index	name	index	name	index	name
0	acq	30	hog	60	platinum
1	alum	31	housing	61	potato
2	barley	32	income	62	propane
3	bop	33	instal-debt	63	rand
4	carcass	34	interest	64	rape-oil
5	castor-oil	35	ipi	65	rapeseed
6	cocoa	36	iron-steel	66	reserves
7	coconut	37	jet	67	retail
8	coconut-oil	38	jobs	68	rice
9	coffee	39	l-cattle	69	rubber
10	copper	40	lead	70	rye
11	copra-cake	41	lei	71	ship
12	corn	42	lin-oil	72	silver
13	cotton	43	livestock	73	sorghum
14	cotton-oil	44	lumber	74	soy-meal
15	cpi	45	meal-feed	75	soy-oil
16	cpu	46	money-fx	76	soybean
17	crude	47	money-supply	77	strategic-metal
18	dfl	48	naphtha	78	sugar
19	dlr	49	nat-gas	79	sun-meal
20	dmk	50	nickel	80	sun-oil
21	earn	51	nkr	81	sunseed
22	fuel	52	nzdlr	82	tea
23	gas	53	oat	83	tin
24	gnp	54	oilseed	84	trade
25	gold	55	orange	85	veg-oil
26	grain	56	palladium	86	wheat
27	groundnut	57	palm-oil	87	wpi
28	groundnut-oil	58	palmkernel	88	yen

Table 14. *Label indexes to names mapping, for the per-label performance graphs for the 20-Newsgroups and Reuters datasets.*