CLUSTERING PARAPHRASES BY WORD SENSE USING TEXTUAL AND VISUAL INFORMATION

Maria Kustikova

A THESIS

in

Robotics

Presented to the Faculties of the University of Pennsylvania in Partial

Fulfillment of the Requirements for the Degree of Master of Science in Engineering

2019

Dr. Chris Callison-Burch
Supervisor of Thesis

Dr. Kostas Daniilidis
Co-Supervisor of Thesis

Dr. Camillo J. Taylor
Graduate Group Chairperson

ACKNOWLEDGEMENT

I would like to thank Dr Chris Callison-Burch, Dr Apidianaki, Dr Daniilidis, Dr Taylor, Dr Cocos, Daphne Ippolito, and Oleh Rybkin for their supervision and advice throughout this project

ABSTRACT

Many Natural Language Processing (NLP) tasks require knowing the sense of polysemous words. Clustering the paraphrases of a word into distinct senses has been used as a step in word sense disambiguation (WSD) algorithms. However, all previous word sense clustering algorithms have relied exclusively on unimodal linguistic features, and in particular, using word representations from distributional semantics. In our work, we incorporate visual features derived from the image search engine into the tasks of word similarity prediction and word sense clustering. Image search engine provide a way to link a query word and a set of top n images for that query word. Based on previous work, n visual or image-based features can be obtained by running n images through a convolutional neural network (CNN) and extracting values from the pre-softmax layer. Following the linguistic approach of having a single representation for a query word, we explore three ways to convert n visual or image-based features into a single representation. For the task of word similarity prediction, we conduct a comprehensive set experiments on thirteen datasets by varying the number of image features n for all three approaches. In addition to comparing different models, we provide a way to combine linguistic and visual features into a multimodal representation by vector concatenation applied with dimensionality reduction. We show that the performance of visual and multimodal representation is comparable to the linguistic representation for some of the datasets. We report that on average the performance increases as n increases. Moreover, we show that sets of image features corresponding to each word also provide a powerful signal for the task of sense clustering, and that by incorporating visual information into our clustering paradigm, we can achieve an alternative sense disambiguation than by using text alone. We report our results on two existing datasets for different part-of-speech (POS) and argue that visual features are better than linguistic features in predicting clusterings for nouns, but are significantly worse for verbs. Finally, we provide limitations of existing datasets, generate a new dataset for word sense, and report our results for different POS.

Contents

List of Tables

List of Figures

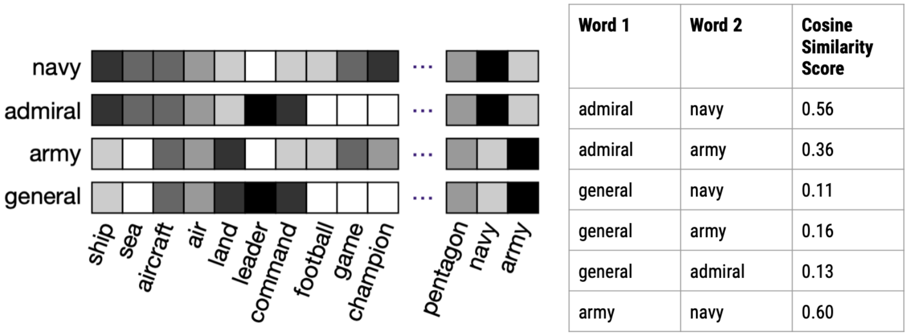
CHAPTER 1 : Introduction

A *polysemous word* is one that has several different meanings or senses. The task of word sense disambiguation (WSD) involves determining the meaning of a word from its surrounding context given a predefined sense inventory. For example, the word ‘bug’ in the context of biology could mean ‘parasite’ or ‘virus’. To a computer scientist, however, ‘bug’ is much more likely to mean ‘error’ or ‘glitch’. WordNet (Miller, 1995) contains manually created sense inventories. It contains the following senses for the noun ‘bug’:

- S1: *bug* (general term for any insect or similar creeping or crawling invertebrate)
- S2: *bug, glitch* (a fault or defect in a computer program, system, or machine)
- S3: *bug* (a small hidden microphone; for listening secretly)
- S4: *hemipterous insect, bug, hemipteran, hemipteron* (insects with sucking mouthparts and forewings thickened and leathery at the base; usually show incomplete metamorphosis)
- S5: *microbe, bug, germ* (a minute life form (especially a disease-causing bacterium); the term is not in technical use)

Many tasks in Natural Language Processing (NLP) like machine translation are incomplete without knowing the sense of a word. Consider the following sentence: ‘The patient is running a fever that seems to be the result of a mild bug’. This cannot be translated accurately from English to French without knowing that the definition of ‘bug’ is most similar to that of ‘virus’, given the context. The task of automatically identifying the senses of a word is knows as word sense induction (WSI).

Hence, there is a need to represent the meaning of a word, as it serves as a prerequisite to many tasks in NLP. One of the traditional approaches to represent the meaning of a word is to use the context in which it appears. The context can be represented as a vector, com-

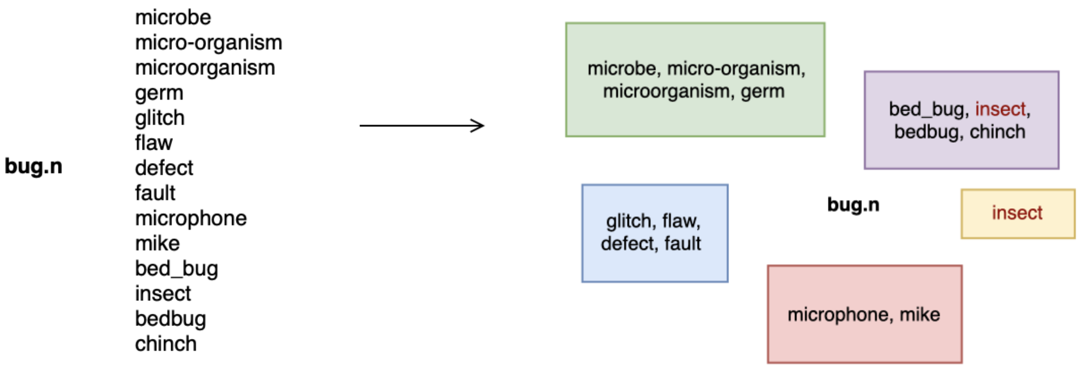


Word 1	Word 2	Cosine Similarity Score
admiral	navy	0.56
admiral	army	0.36
general	navy	0.11
general	army	0.16
general	admiral	0.13
army	navy	0.60

Figure 1: An abstract representation of word vectors that represent a word using the contexts in which it appears under the assumption of the distributional hypothesis

monly referred to as a word embedding. The key idea relies on the distributional hypothesis (Harris, 1954), which tells that words that appear in similar context have similar vectors and similar semantics. This representation allows to compare words via multiple vector similarity metrics, for instance cosine similarity. Figure 1 denotes an abstract representation of word embeddings along with the cosine similarity scores for a subset of those embeddings. It can be seen that words ‘army’ and ‘navy’ have a relatively high similarity score, whereas ‘general’ and ‘navy’ have a relatively low similarity score. This is most likely justified by the word ‘general’ being a polysemous word that could appear both as an adjective and as a noun.

In our work, we focus on two tasks: word similarity prediction and clustering paraphrases by word sense. The first task asks to predict a similarity and/or relatedness score between a pair of words. For instance, in Figure 1 how similar are ‘navy’ and ‘general’ or how similar are ‘admiral’ and ‘general’? The second task is to group paraphrases by word sense. The task is summarised in Figure 2 and can be formulated as follows: given a query term and a set of paraphrases for that query term, create a clustering such that a cluster in this clustering represents a distinct sense.



(a) Input (b) Output

Figure 2: WordNet+ Gold 2.0 Dataset gold clustering for the word *bug*. The objective is to cluster paraphrases such as *bug* into its different senses Cocos and Callison-Burch (2016)

The approach most closely related to our work is that of Cocos and Callison-Burch (2016) which explores advanced clustering algorithms and similarity measures for clustering paraphrases by word sense. A key component to clustering is choosing how to define the similarity, or affinity, between two data points. Cocos and Callison-Burch (2016) experimented with several text-based measures of affinity between two paraphrases, such as second-order paraphrases and distributional semantics. Most clustering algorithms require a similarity matrix as an input. A similarity matrix for a paraphrase set of n terms is $n \times n$ symmetric matrix of non-negative values where each element gives a pairwise similarity score. Therefore, the first task of similarity prediction is of great importance to the second task of sense clustering.

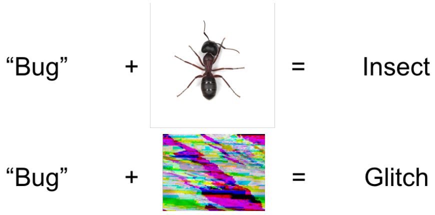


Figure 3: The presence of an image can provide a useful, and at times complimentary signal for several NLP tasks

In our work, we show that images provide a useful, and at times complementary signal to text for performing word similarity prediction and word sense induction. Image features have been used in other NLP tasks like learning translations via visual similarity of words (Bergsma and Van Durme, 2011). Images intuitively provide a level of information that is often helpful in NLP tasks. For example, to disambiguate the meaning of *bug*, the presence of a corresponding image can help with the understanding of the sense of a word, as can be seen in Figure 3. Images search engines provide a way to associate a set of images with a query word or phrase. Figure 4 shows top 5 images for some of the paraphrases seen in Figure 2 collected by Callahan (2017) from the Google Image Search.

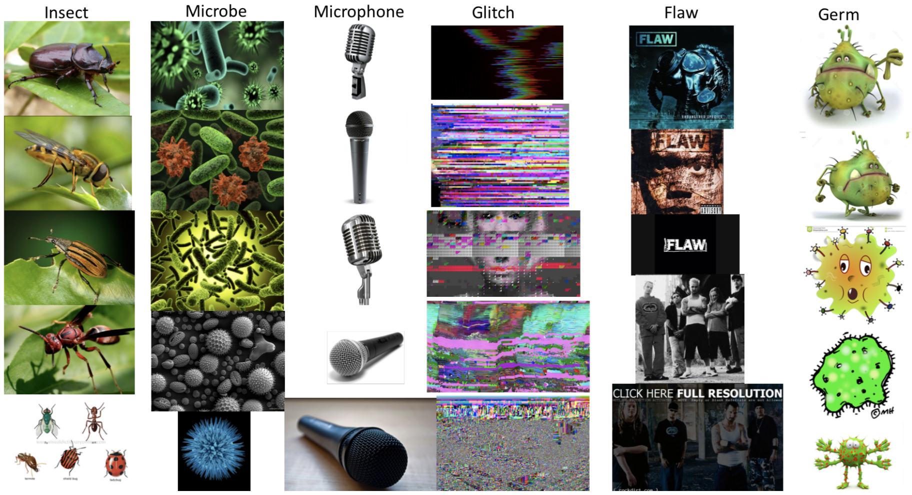


Figure 4: Top-5 images from the collected dataset (Callahan, 2017) for the paraphrases observed in Figure 2

We propose a new method to cluster paraphrases based on word sense using their textual as well as image features. Our goal is to determine the best way to leverage the information provided by images to accurately predict the senses of a word given its paraphrases. We conduct a broad range of experiments to address the following research questions:

- Can we create vector representations using images instead of text?

- Can image representations be used instead of text for word similarity and sense clustering? Can image representation consistently help capture the different senses of a word?

- What would be the best way to combine similarity measures from multiple modes when clustering paraphrases?

- Does image representation work better for a particular part-of-speech (POS)?

- Can image concreteness values augment sense clustering? *Concreteness* is a concept from psycholinguistics indicates the degree to which the concept denoted by a word refers to a perceptible entity (Brysbaert et al., 2013).

The main contributions of our word are outlined below:

- Extend the results of word similarity prediction on a variety datasets

- Experiment with the number of image features, dimensionality and ways to combine image features to represent a word

- Provide a novel approach to perform clustering paraphrases by word sense using images

- Provide a novel approach to perform clustering paraphrases by word sense by combining multimodal data coming from images and text

- Breakdown evaluation by POS

- Generate a new dataset for clustering paraphrases by word sense that originates from WordNet+ dataset

This thesis is organised in the following way: Chapter 2 describes approaches to incorporate visual features into a word representation from images retrieved from search engines. Chapter 3 provides an overview, literature review, dataset descriptions, experiments and discussion for the task of predicting similarity of word pairs. Chapter 4 provides an overview, literature review, dataset descriptions, experiments and discussion for the task of clustering paraphrases by word sense. The chapter also focuses on limitations of the existing datasets and generation of the new dataset. Finally, Chapter 5 provides a conclusion and discussion for future work.

CHAPTER 2 : Incorporating visual features

2.1. Overview

This chapter provides a literature overview and approaches in using information from image search engines for a variety of tasks, and describes several approaches taken for this thesis research as a result of previous work.

2.2. Approach

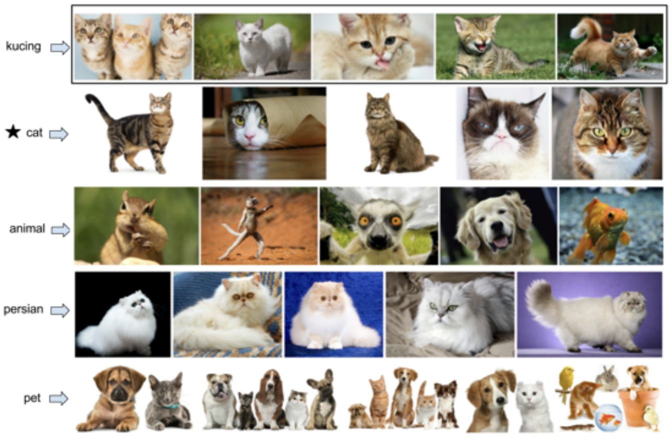


Figure 5: Bergsma and Van Durme (2011) approach of learning translations via visual similarity. The top row contains five images for the Indonesian word *kucing*. The bottom 4 rows display top 4 translations in English. Figure taken from Hewitt et al. (2018)

We build on data created by Callahan (2017) that was used by Hewitt et al. (2018). Callahan (2017) re-created the experiments of Bergsma and Van Durme (2011) at a much larger scale. Bergsma and Van Durme (2011) learn translations via visual similarity of images associated with words in foreign languages. They retrieve images for a foreign word, convert them to vectors via SIFT and colour histogram features, and then compare the foreign words vector against vectors representing all English words. Figure 5 illustrates this approach for an Indonesian word 'kucing' and top 4 English translations with 'cat' being the most similar word to 'kucing' based on visual similarity.

The collected dataset by Callahan (2017) contains 100 images for around 10000 words in each of 100 foreign languages. The dataset also contains images and text from the web pages where an image appeared at for each of their translations into English. In total, the dataset contains 35 million images and web pages. In this setup, the word serves as a query to the data that originates from an image search engine. Since the data originates from Google Image Search, there is a question about how well a set of images represent a query word since search engines automatically associate words and images. Hewitt et al. (2018) show that on average 86% of images were evaluated by Amazon Mechanical Turk workers to be good representations of their target word. For vocabulary words that occur in datasets used in predicting similarity of word pairs seen in Chapter 3 and clustering paraphrases using word sense seen in Chapter 4, the average number of images per word is around 90. We chose to collect at most 100 images following the set up of Hewitt et al. (2018) and contrary to the suggestions on the optimal number of images of between 10 and 20 proposed by Kiela et al. (2016) in order to perform systematic evaluation on our side.

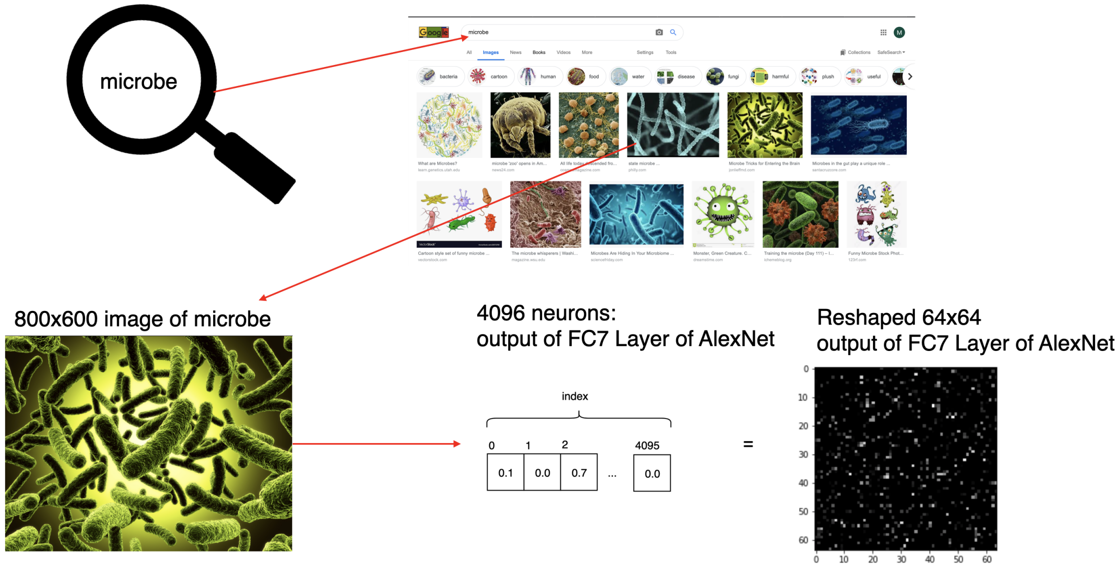


Figure 6: An example of how n image features are generated for a word ‘microbe’

Kiela et al. (2015) showed that using CNN features is superior to using SIFT and colour histograms for vector generation used by Bergsma and Van Durme (2011). As such, we follow the setup of Kiela et al. (2015), where for each image extracted for a word in vocabulary, we extract 4096-dimensional features from the FC7 layer of a CNN called AlexNet trained on ImageNet (Krizhevsky et al., 2012). The FC7 layer is the fully connected 7th layer and is the second to last layer in the AlexNet architecture. The process of retrieving top images for a word 'microbe' and converting an image to a 4096-dimensional feature is illustrated in Figure 6. Figure 7 shows an abstract representation of n image features for a given word.

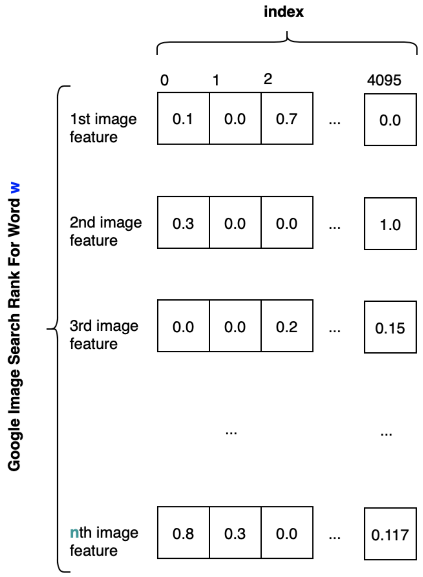


Figure 7: Abstract representation of n image features for a given word w

2.3. Using Multiple Images As The Representation For A Single Word

Since we have up to 100 vectors derived from the 100 images retrieved for each word, we need a way of combining the vectors into a single representation of a word, or combining them to make a comparison between a pair of words. The subsequent subsections describe multiple approaches taken to compute similarity using extracted image features for a pair of words.

Let us denote the following:

- w = word
- n = number of image features for a given word w
- $\mathcal{I}(w)$ = set of image features for a given word w

2.3.1. Average Similarity for a Word

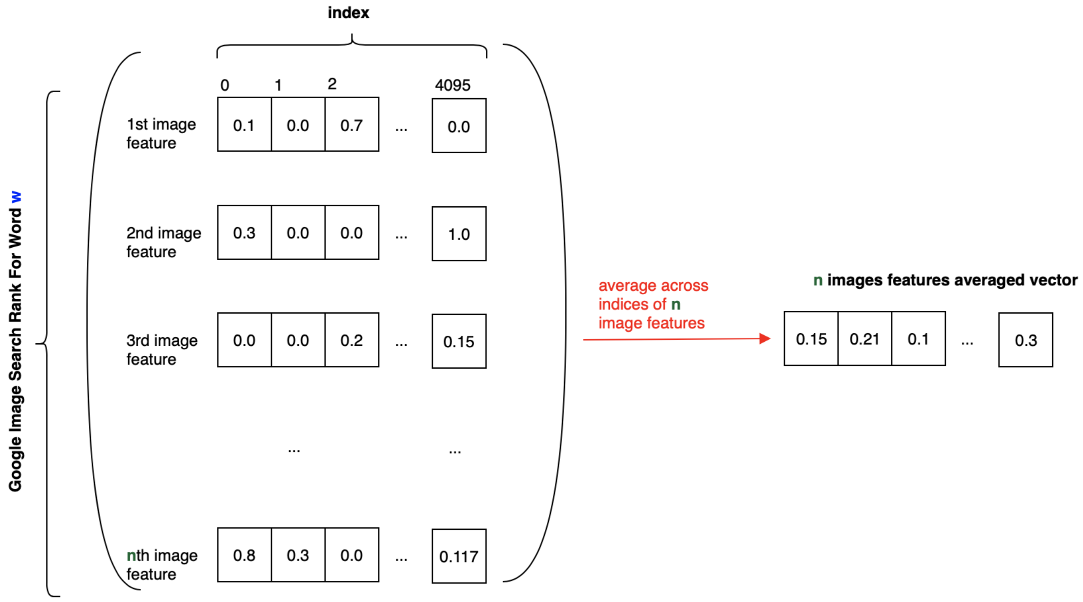


Figure 8: An average of n image features can be computed computed for to represent a single word w

In this approach, we take n image features for a word w and produce a single 4096-vector containing the average of each column as seen in Equation 2.1.

$$\text{VECAVG}(w) = \frac{1}{n} \sum_{i \in \mathcal{I}(w)} i \qquad (2.1)$$

A diagram representing the VECAVG operation is given in Figure 8. Hence, this approach produce a vector embedding for a word w by averaging n image features. Computing similarity between a pair of words w_1 and w_2 then becomes a similarity between their two vector representations as seen in Equation 2.2.

$$\textsc{Avg}(w_1, w_2) = \text{sim}(\text{VECAVG}(w_1), \text{VECAVG}(w_2)) \tag{2.2}$$

For SIM a similarity metric, for instance, cosine similarity can be used to calculate similarity between the averaged high-dimensional vectors. Cosine similarity as been shown to be a very effective measure on many semantic benchmarks (Bullinaria and Levy (2007), Padó and Lapata (2007)). Kiela and Bottou (2014) and Kiela et al. (2015) refer to this approach of averaging n images as CNN-MEAN.

2.3.2. Average Maximum Similarity Between a Pair of Words

In this approach, we compute the average maximum similarity between the image feature sets to compute the similarity between a pair of words w_1 and w_2 as seen in Equation 2.3.

$$\textsc{AvgMax}(w_1, w_2) = \frac{1}{n_1} \sum_{i_1 \in \mathcal{I}(w_1)} \max_{i_2 \in \mathcal{I}(w_2)} (\text{sim}(i_1, i_2)) \tag{2.3}$$

This measure was originally introduced for the task of translating words between English and a foreign language by looking only at corresponding images (Bergsma and Van Durme, 2011; Kiela et al., 2015). Kiela et al. (2015) found AvgMax to be the best-performing model in comparison to MaxMax or CNN-MEAN.

Figure 9 provides an abstract representation of how an $\textsc{AvgMax}(w_1, w_2)$ is computed. It can be seen that the first step is to compute maximum pairwise similarity of each one of image features for w_1 to image features for w_2. The second step then averages the maximum pairwise similarity computed and dividing by the number of image features for w_1.

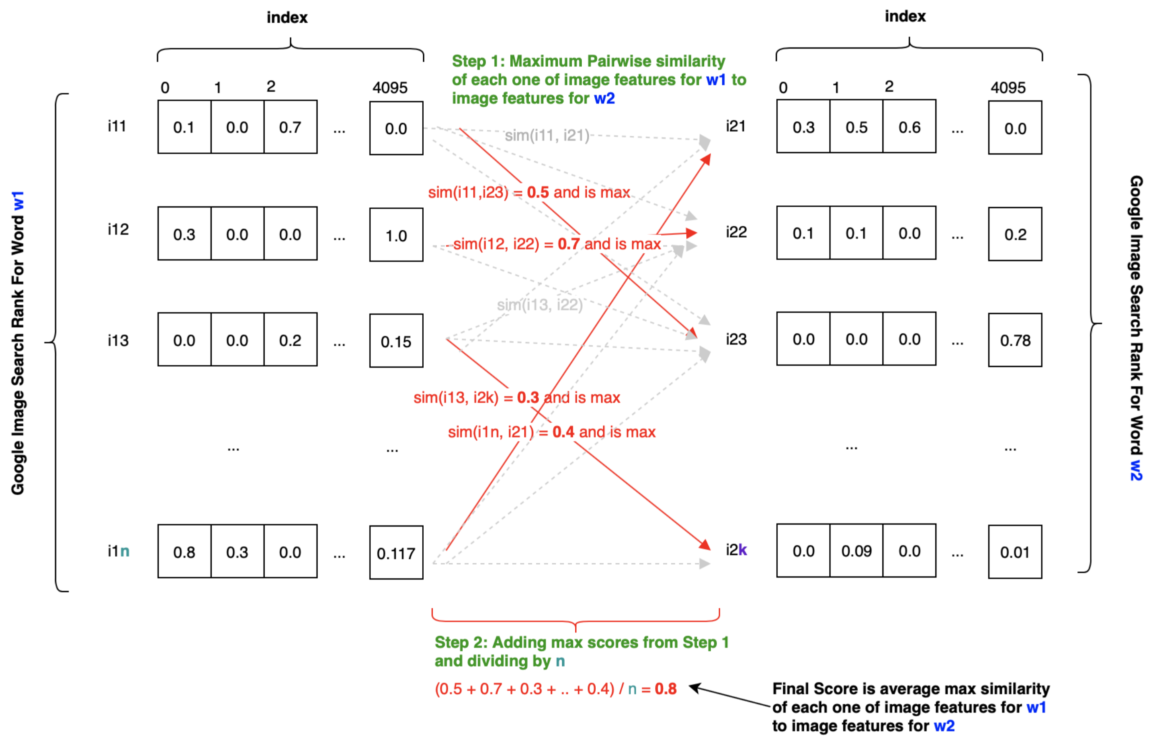

index
0
1
2
4095
Google Image Search Rank For Word w1
i11
0.1
0.0
0.7
...
0.0
i12
0.3
0.0
0.0
...
1.0
i13
0.0
0.0
0.2
...
0.15
...
...
i1n
0.8
0.3
0.0
...
0.117
Step 1: Maximum Pairwise similarity of each one of image features for w1 to image features for w2
sim(i11, i21)
sim(i11,i23) = 0.5 and is max
sim(i12, i22) = 0.7 and is max
sim(i13, i22)
sim(i13, i2k) = 0.3 and is max
sim(i1n, i21) = 0.4 and is max
index
0
1
2
4095
i21
0.3
0.5
0.6
...
0.0
i22
0.1
0.1
0.0
...
0.2
i23
0.0
0.0
0.0
...
0.78
...
...
i2k
0.0
0.09
0.0
...
0.01
Google Image Search Rank For Word w2
Step 2: Adding max scores from Step 1 and dividing by n
(0.5 + 0.7 + 0.3 + .. + 0.4) / n = 0.8
Final Score is average max similarity of each one of image features for w1 to image features for w2

Figure 9: Abstract representation of how an AvgMax(w_1, w_2) is computed

The value produced in Equation 2.3 might not be necessarily symmetric, for example, if two images for w_1 have the same closest image in the set of w_2 or when the number of image features for the first word might not be equal to the number of images features for the second word. In order to demonstrate such case let us use integers instead of vectors as the elements in each image set. Let: $\mathcal{I}_1 = (1, 2)$ and $\mathcal{I}_2 = (1, 5, 9)$. The average maximum pairwise similarity of $\mathcal{I}_1$ to $\mathcal{I}_2$ is then: $\frac{\max(\text{sim}(1,1),\text{sim}(1,5),\text{sim}(1,9))+\max(\text{sim}(2,1),\text{sim}(2,5),\text{sim}(2,9))}{2} = \frac{\text{sim}(1,1)+\text{sim}(2,1)}{2}$. The average maximum pairwise similarity of $\mathcal{I}_2$ to $\mathcal{I}_1$ is equal to the following: $\frac{\max(\text{sim}(1,1),\text{sim}(1,2))+\max(\text{sim}(5,1),\text{sim}(5,2))+\max(\text{sim}(9,1),\text{sim}(9,2))}{2} = \frac{\text{sim}(1,1)+\text{sim}(5,2)+\text{sim}(9,1)}{3}$. Therefore it can be seen that the average maximum pairwise similarity of $\mathcal{I}_1$ to $\mathcal{I}_2$ and average maximum pairwise similarity of $\mathcal{I}_2$ to $\mathcal{I}_1$ are not equal to each other. Since most clustering algorithms require the input affinity matrix to be symmetric, we use the following symmetric form as seen in Equation 2.4.

$$\textsc{SymAvgMax}(w_1, w_2) = \frac{\textsc{AvgMax}(w_1, w_2)}{2} + \frac{\textsc{AvgMax}(w_2, w_1)}{2} \tag{2.4}$$

As seen in Section 2.3.1 we can use a measure such as cosine similarity to measure similarity between a pair of vectors.

2.3.3. Average Average Similarity Between a Pair of Words

We replicate the approach Section 2.3.2, but instead of using average maximum to compute the similarity between a pair of words w_1 and w_2, we use average of average to compute the similarity between a pair of words as seen in Equation 2.5. The motivation behind using average of average came from the lack of occurrence of such measure in the previous work.

$$\textsc{AvgAvg}(w_1, w_2) = \frac{1}{n_1} \sum_{1 \in \mathcal{I}(w_1)} \mathrm{avg}_{i_2 \in \mathcal{I}(w_2)}(\mathrm{sim}(i_1, i_2)) \tag{2.5}$$

2.4. Other Related Work

In this section, we describe other NLP work that uses multimodal data obtained from image search engine. Bergsma and Van Durme (2011) were one of the first researchers to perform word-to-word translations using multimodal data obtained from image search engine. They used this monolingual connection between a word and an image to learn bilingual translations on 15 language pairs based on whether the corresponding images have resembling visual features. Bergsma and Goebel (2011) used image to word connection to help predict lexical selectional preferences. In particular, the area of their focus lied on predicting whether a noun argument occurs as the direct object of a verb predicate. In order to do so, for each verb-noun pair they retrieved images of a noun, extracted visual information from images and then used a model on those visual features to output a plausibility score for a verb-noun pair. In addition, Bergsma and Goebel (2011) demonstrated that Google image search yield representations of higher quality when compared to Flickr.

Bruni et al. (2014) proposed an architecture for integrating text and image-based distributional information that is superior to predicting semantic similarity and relatedness for a pair of words. Kiela and Bottou (2014) presented a novel approach in constructing multimodal representations by combining a Skip-gram linguistic representation vector and a visual concept as an extracted layer of a deep CNN trained using ImageNet (Krizhevsky et al., 2012) or ESP Game (von Ahn and Dabbish, 2004). Kiela and Bottou (2014) applied this vector representation into semantic relatedness evaluation tasks and outperformed representations that are linguistic or standard bag-of-visual-words (BoVW) (Sivic and Zisserman, 2003). Inspired by the traditional bag-of-words BoW method, BoVW gets a visual representation from an image by connecting each of its local descriptors to a cluster histogram with a use of a clustering algorithm. In that year Kiela et al. (2014) published another paper on improving the results of multimodal representation by deciding whether to include perceptual input for a concept or not based on concreteness. Kiela et al. (2015) used image search engine along with the query word for lexical entailment detection, and in particular examining generality of the hypernym compared to the hyponym based on their related images. Kiela et al. (2015) published another paper on using image features obtained from the pre-softmax layer of CNNs for the task of bilingual lexicon induction. The authors argued that the reason for choosing CNN-derived image representations was that in comparison to traditional bag of visual models used in multimodal distributional semantics (Bruni et al. (2014), Kiela and Bottou (2014)), this representation yields higher quality representations. Furthermore, Kiela et al. (2015) experimented with various visual similarity metrics between two sets of n images and the corresponding features.

The following year, a variety of NLP studies used the CNN-derived representation from top-10 images in Google image search proposed by Kiela and Bottou (2014) in conjunction with linguistic representation: Shutova et al. (2016) performed metaphor identification, Bulat et al. (2016) obtained property norms, predictions, while Vulić et al. (2016) created bilingual multimodal embeddings to perform bilingual lexicon learning. Lazaridou et al. (2015) extended the Skip-gram model of Mikolov et al. (2013) by taking visual representation and evaluating against a variety of semantic benchmarks. Following the success of multimodal representation learning in a range of tasks, Kiela (2016) developed a MMFeat toolkit for obtaining feature representations for visual information.

Kiela et al. (2016) performed an evaluation in comparing CNNs architectures of state-of-the-art models, explored raw input images coming from various engines, and identified the optimal number of images. The approach proposed by Kiela and Bottou (2014) in using deep CNNs trained on Google Images for visual groundings has also been applied for decoding brain activity. Anderson et al. (2017) used both the linguistic representation and CNN-derived image representation in isolation for decoding brain activity, and in particular, decoding abstract nouns. Anderson et al. (2017) observed that the former representation yields greater accuracies for abstract nouns, however the performance of both models is similar for more concrete nouns. In another study Bulat et al. (2017) used MMFeat toolkit (Kiela, 2016) to perform a systematic evaluation of text-based, image-based and multimodal semantic models in their ability to predict patterns of conceptual representation in the human brain.

Glavaš et al. (2017) presented research in semantic text similarity, which measures semantic equivalence between short texts, using the MMFeat toolkit (Kiela, 2016). The authors used Bing image search with 20 images per word and VGGNet (Simonyan and Zisserman, 2014) pre-trained on the ImageNet classification task (Russakovsky et al., 2015) to extract visual representation. Glavaš et al. (2017) found that multimodal representation achieves the best performance than visual and linguistic measures in isolation.

Bhaskar et al. (2017) performed a comparison between textual, visual and combined modalities for distinguishing between abstract and concrete nouns following the feature extraction and suggestions of Kiela et al. (2016) and querying up to 25 images per word. While the predictions achieved high performance, the authors found that the difference between unimodal and multimodal representations in terms of performance was negligible. In another study Hartmann and Søgaard (2018) argue that the previous work on bilingual lexicon induction with multimodal representation only applies to nouns and does not scale to other part-of-speech (POS), for instance adjectives and verbs. Hewitt et al. (2018) address the challenges of translation outlined by Hartmann and Søgaard (2018) by finding that images are just as effective for translating more complex phrases than simple nouns, and expanding the dataset for over 260000 English words and 32 foreign languages.

Wang et al. (2018) proposed three novel methods for building a multimodal model that can fuse the semantic word representation from various modalities according to different types of words by obtaining visual representation from averaging CNN-derived representation. Collell et al. (2017) proposed an integration between language and vision that provides a way to 'imagine' missing visual information and a method to build a multimodal representation with the use of mapped vectors. Collell and Moens (2018) uncovered that the multimodal mappings from the CNNs can produce mapped vectors more similar to the input than to the target with respect to the semantic structure. They proposed a new similarity measure that explicitly quantifies similarity between the neighbourhood structure of two sets of vectors. Kiros et al. (2018) published a large-scale lookup operation called Picturebook. The Picturebook extracts top images from Google image search and extracts image embeddings by feeding them into CNN. Kiros et al. (2018) report result across a range of NLP tasks: similarity and relatedness between a pair of words, natural language inference, sentiment or topic classification, image-sentence ranking and machine translation. The main contributions of their research are in the collection of word representations from GloVe (Pennington et al., 2014), which is orders of magnitude larger than previous work, and in multimodal gate mechanism for choosing between GloVe and Picturebook that can be applied in a task-dependent way.

CHAPTER 3 : Predicting Similarity of Word Pairs

3.1. Overview

For this part of the research, we seek to understand how well visual-based word representations measure semantic similarity and relatedness between a pair of words. The motivation behind conducting this set of experiments is that the similarity score between a pair of words produced by our best word representation is used directly as an input into a similarity matrix in the clustering algorithm described in Chapter 4. Moreover, knowledge from these experiments provides an insight into understanding whether our image-based word representations are comparable to existing state-of-art text-based vector representations. As importantly, these experiments allow us to identify the type of words for which our approach is well-suited. This chapter contains a literature review on conventional and multimodal distributional semantic models, descriptions of the datasets, evaluation metrics, along with the experiments and results.

3.2. Literature Review

3.2.1. Distribution semantic models

Before the appearance of the multimodal representation, the task of predicting similarity between a pair of words was approached using traditional distribution semantic models (DSMs) that rely only on linguistic (unimodal) environment. DSMs are based on the distributional hypothesis (Harris, 1954) that states that words are likely to be semantically related if they occur in similar contexts (Bruni et al., 2014). The most common type of DSMs is semantic space models commonly referred to as vector space models or word embeddings, which approximate the meaning of words with vectors that record the distributional history in a corpus (Turney and Pantel, 2010). DSMs have been extremely effective in a variety of tasks in semantics like semantic composition and analogical mapping (Clark, 2015; Turney and Pantel, 2010; Mikolov et al., 2013)).

Word2vec is one of the most popular word embeddings techniques. Word2vec was developed by Mikolov et al. (2013) and consists of two neural network language models: continuous bag-of-words (CBOW) and Skip-gram. Both models are trained on words inside a window of a pre-defined length, which is moved along the corpus. CBOW model predicts a word given a surrounding window of context words, while a Skip-gram model predicts the surrounding window of context words, given a word. Another commonly used word embedding is called GloVe (Pennington et al., 2014). Like word2vec, GloVe is an unsupervised approach based on the distributional hypothesis. FastText word representations (Joulin et al., 2016) is an extension and improvement of word2vec, which allows to compute word representations for out-of-vocabulary (OOV) words with a use of character n-grams. Recently, ELMo word representation (Peters et al., 2018) have improved state-of-the-art across a range of NLP tasks, one of which includes sentiment analysis. In ELMo, an embedding of a word is computed from the the internal states of a deep bidirectional language model (LM) pre-trained on a large corpus.

An alternative implementation of DSMs called probabilistic topic models have been explored in the literature (Griffiths et al., 2007). The similarity between probabilistic models and DSMs is that they also gather co-occurrence information from corpus, but the difference between the two approaches is that probabilistic models have an assumption about words in corpus having a probabilistic structure. In probabilistic model words are a probability distributions over a set of topics and can define semantic meaning between a pair of words using inference.

3.2.2. Multimodal Distribution Semantic Models

The conventional distributional semantic models lack visual information that could be extracted from the physical world. This observation gave rise to the development of multimodal distributional semantic models that combines data originating from two modalities: linguistic and perceptual. A literature review on extracting visual information from image search engine and applying a multimodal representation to a variety of NLP tasks was a provided in Section 2.4. When describing the applications of multimodal representation in NLP tasks, a few of the research studies in predicting word pair similarity were mentioned in that section, such as the research done by Kiela and Bottou (2014), Bruni et al. (2014), Kiela et al. (2016), Collell et al. (2017), and Kiros et al. (2018).

In the context of word similarity, Kiela and Bottou (2014) proposed a novel model of multimodal representation that combines Skip-gram model of Mikolov et al. (2013) trained on Wikipedia (400M) and British National Corpus (100M) with image features extracted from the deep CNN trained on ImageNet (Krizhevsky et al., 2012) or ESP Game dataset (von Ahn and Dabbish, 2004). In order to combine linguistic and visual features, authors concatenated the centred and L2-normalized feature vectors that were learned independently from each other. The authors reported results on 2 datasets described in the next sections: WordSimilarity-353 (Finkelstein et al., 2001) and MEN-3000 (Bruni et al., 2014). Kiela et al. (2014) defined an unsupervised method called image dispersion to distinguish abstract from concrete words based on the observation that average cosine distance between all the visual representations of a word negative correlates with its concreteness.

The same year Bruni et al. (2014) proposed a similar pipeline as Kiela and Bottou (2014) of training visual and linguistic representation independently, however for linguistic representation Bruni et al. (2014) used semantic space model called Hyperspace Analog to Language model (HAL) (Lund and Burgess, 1996) to determine a window of context words. Bruni et al. (2014) reported various 'fusion' methods for combining visual and linguistics features by first using concatenation and then applying feature and scoring fusion functions, essentially applying the Singular Value Decomposition (SVD) (Golub and Reinsch, 1970) to concatenated vectors. Silberer and Lapata (2014) used a different visual representation that were annotated with high-level of visual attributes, and proposed a more complex multimodal fusion strategy based on stacked auto-encoders. The idea behind this approach is that the encoder is given both linguistic and perceptual features, from which multimodal embeddings arise from the hidden representation.

Lazaridou et al. (2015) extended a Skip-gram model of Mikolov et al. (2013) to support multimodal representation by their skip-gram model as maximal margin objective function that tries to minimise the distance between the two vectors. Essentially, in comparison to previous work, the authors built multimodal representations with raw inputs of both linguistic and visual information. As before, the evaluation was performed on set on MEN-3000, SimLex-999 (Hill et al., 2015) and a few other datasets. Kiela et al. (2016) performed a systematic evaluation in comparing three CNNs architecture, exploring multiple image retrieval engines, and explored the optimal number of images. The experiments were performed on predicting semantic similarity between a pair of words. Kiela et al. (2016) found that the performance across AlexNet trained on ImageNet (Krizhevsky et al., 2012), GoogleNet (Szegedy et al., 2015) and VGGNet (Simonyan and Zisserman, 2014) is similar and proposed usage of AlexNet and VGGNet for overall best performance. Moreover, Kiela et al. (2016) found both Google and Bing to be suited to perform full-coverage experiments, meaning that when highly abstract words are included, there is no negative image on the method's performance. Lastly, the optimal number of images based on their systematic evaluation is between 10 and 20 images, since the performance of a model stabilises around 10 images for Google and Bing as the data source.

Collell et al. (2017) proposed a new approach that uses a feed-forward neural network to learn a mapping between visual and text modalities, which are directly used to build the multimodal representations. Collell et al. (2017) found that in the process of mapping an irrelevant visual information is discarded, hence, improving the performance on various datasets. Some of the datasets that Collell et al. (2017) used are MEN-3000 along with WordSimilarity-353-REL Agirre et al. (2009) to measure general relatedness and SimLex-999 for measuring semantic similarity.

Wang et al. (2018) presented three novel fusion methods that combine modalities according to different types of words by assigning importance weights to each modality. The weights are learned using the weak supervision of word association pairs. The proposed method outperformed previous state-of-the-art multimodal as well as the traditional linguistic representations on the datasets mentioned before. Wang et al. (2018) used Glove vectors for representation. Similarly to Wang et al. (2018), Kiros et al. (2018) presented a framework that fuses GloVe embeddings with visual representation obtained from the search engine that were fed into the CNN. Although the representation presented by Kiros et al. (2018) focused on other applications in NLP, their experiments on word similarity predictions explored the optimal number of images and concreteness scores for the SimLex-999.

3.3. Word Similarity Data Sets

As described in Section 3.2, there is a large number of existing lexical semantics evaluation benchmarks available to evaluate semantic similarity and relatedness between word pairs in English. It is important to note that there is a discrepancy between existing gold standard datasets, since some do not specify or clearly distinguish between similarity and relatedness or association. Furthermore, the literature review for multimodal representations does not clearly differentiate between semantic similarity and relatedness. We chose to perform a system evaluation on all 13 dataset benchmarks, even though for the visual-based features we hypothesise that the semantic similarity measure is of greater importance in comparison to relatedness. The datasets used for evaluation are described in the subsequent sections.

3.3.1. RG-65 and MC-30

RG-65 dataset released by Rubenstein and Goodenough (1965) in 1965 contains 65 noun word pairs evaluated by the human judgement. The similarity of each pair in the dataset has a real value between 0 and 4, where the higher the value, the higher the similarity of meaning between a word pair is. This dataset measures similarity between a pair of words rather than relatedness. Rubenstein and Goodenough (1965) report the inter-annotator agreement to be $r = 0.85$ of Pearson correlation. Table 1 contains examples from top 10 and bottom 10 of this dataset. MC-30 dataset by Miller and Charles (1991) is a dataset with 30 noun word pairs taken from RG-65 dataset. MC-30 dataset has a wider use by the research community to assess the semantic similarity of words.

Word 1	Word 2	Score
gem	jewel	3.94
midday	noon	3.94
automobile	car	3.92
cemetery	graveyard	3.88
cushion	pillow	3.84
boy	lad	3.82
⋮	⋮	⋮
automobile	wizard	0.11
autograph	shore	0.06
fruit	furnace	0.05
noon	string	0.04
rooster	voyage	0.04
chord	smile	0.02

Table 1: Selected examples from top 10 and bottom 10 of RG-65 dataset

3.3.2. WordSimilarity-353

The WordSimilarity-353 Test Collection[1] released by Finkelstein et al. (2001) contains 353 pairs of noun words in English along with the mean score of human-assigned similarity judgements. The subjects of experiments were asked to give a score of relatedness of the words in pairs on a scale from 0 to 10, where 0 indicates totally unrelated words and 10 indicates very much related or identical words. Based on the given instructions this dataset measures association or relatedness between words and not similarity. The inter-annotator agreement reported for this dataset is $\rho = 0.611$ of Spearman correlation coefficient. Table 2 displays selected examples from top 10 and bottom 10 of this dataset according to the human judgements. There is an overlap of 30 word pairs between the WordSimilarity-353 and RG-65 dataset.

Word 1	Word 2	Score
tiger	tiger	10.00
journey	voyage	9.29
midday	noon	9.29
money	cash	9.15
coast	shore	9.10

[1] http://www.cs.technion.ac.il/~gabr/resources/data/wordsim353/

football	soccer	9.03
⋮	⋮	⋮
monk	slave	0.92
sugar	approach	0.88
noon	string	0.54
chord	smile	0.54
professor	cucumber	0.31
king	cabbage	0.23

Table 2: Selected examples from top 10 and bottom 10 of WordSimilarity-353 dataset

3.3.3. WordSimilarity-353-SIM and WordSimilarity-353-REL

The WordSimilarity-353 described in Section 3.3.2 was further split into two subsets[2] by Agirre et al. (2009), where the first subset measures similarity (WordSimilarity-353-SIM, referred in the literatures as WS-SIM or Wordsim) and the second subset measures relatedness (WordSimilarity-353-REL). Agirre et al. (2009) argued that there is a difference between similarity and relatedness, and WordSimilarity-353 does not distinguish between the two relations. In order to split the dataset, they classified each pair of words based on WordNet (Miller (1995)) data as one of the following:

- similar pairs (synonyms, antonyms, identical, hyponym-hypernym)
- related pairs (meronym-holonym, score > 5)
- unrelated pairs (none of the above, score ≤ 5)

The WordSimilarity-353-SIM was then created as the union of all similar and unrelated pairs, whereas WordSimilarity-353-REL was the union of related and unrelated pairs. One of their keys observations when comparing the performances of models for the divided dataset was that two words are similar if their synsets are close in the hierarchy of WordNet, and two words are related if there is a connection between them in the hierarchy. The overall number of pairs in WordSimilarity-353-SIM is 203 and the overall number of pairs in WordSimilarity-353-REL is 252. The reported inter-annotator agreement reached $\rho = 0.667$

[2] `http://alfonseca.org/eng/research/wordsim353.html`

of Spearman correlation for WordSimilarity-353-SIM and $\rho = 0.72$ for WordSimilarity-353-REL. Tables 3 and 4 displays selected examples top 10 and bottom 10 of these two datasets. As can be seen from these tables, there is an overlap in the bottom 10 examples, which can be explained by the method the words were split up.

Word 1	**Word 2**	**Score**
tiger	tiger	10.00
journey	voyage	9.29
midday	noon	9.29
money	cash	9.15
coast	shore	9.10
football	soccer	9.03
⋮	⋮	⋮
monk	slave	0.92
sugar	approach	0.88
noon	string	0.54
chord	smile	0.54
professor	cucumber	0.31
king	cabbage	0.23

Table 3: Selected examples from top 10 and bottom 10 of WordSimilarity-353-SIM datasets

Word 1	**Word 2**	**Score**
environment	ecology	8.81
Maradona	football	8.62
OPEC	oil	8.59
computer	software	8.50
money	bank	8.50
Jerusalem	Israel	8.46
⋮	⋮	⋮
monk	slave	0.92
sugar	approach	0.88
noon	string	0.54
chord	smile	0.54
professor	cucumber	0.31
king	cabbage	0.23

Table 4: Selected examples from top 10 and bottom 10 of WordSimilarity-353-REL datasets

Tables 5 and 6 displays two tables with pairs of words and the corresponding scores for words that are Similar but not Related and words that are Related but not Similar based on the differences between the WordSimilarity-353 Similarity and Relatedness datasets. It can be seen from that that some of the word pairs in Table 5 are clear antonyms, for instance ‘life’ and ‘death’ or ‘king’ and ‘queen’, and other word pairs follow type-of relationship common for when describing hypernym-hyponym, for instance ‘aluminium’ is a type of ‘metal’ or ‘water’ is a type of ‘liquid’. Table 6 displays words that are not similar, but related. For instance ‘popcorn’ and ‘movie’ share association since popcorn is a snack commonly eaten in front of a movie.

Word 1	**Word 2**	**Score**
seafood	lobster	8.70
king	queen	8.58
championship	tournament	8.36
Harvard	Yale	8.13
liquid	water	7.89
life	death	7.88
aluminum	metal	7.83
Mexico	Brazil	7.44
tiger	cat	7.35
physics	chemistry	7.35
street	place	6.44
train	car	6.31
bread	butter	6.19
glass	metal	5.56
cup	artifact	2.92

Table 5: Selected examples from WordSimilarity-353-SIM that do not occur in WordSimilarity-353-REL

Word 1	**Word 2**	**Score**
weather	forecast	8.34
bank	money	8.12
stock	market	8.08
closet	clothes	8.00
admission	ticket	7.69
drug	abuse	6.85
competition	price	6.44
production	crew	6.25
movie	popcorn	6.19
announcement	warning	6.00
game	round	5.97
baseball	season	5.97
journey	car	5.85
territory	surface	5.34
credit	information	5.31

Table 6: Selected examples from WordSimilarity-353-REL that do not occur in WordSimilarity-353-SIM

3.3.4. MTurk-287

The MTurk-287 dataset released by Radinsky et al. (2011) contains 287 word pairs that were constructed using Amazon Mechanical Turk (AMT) to get the human similarity scores. Those 287 word pairs are generate from the New York Times papers, do not overlap with RG-65 or WordSimilarity-353 datasets, and are on a scale from 1 to 5. Table 7 contains selected examples from top 10 and bottom 10 of this dataset according to the human judgements.

Word 1	**Word 2**	**Score)**
funeral	death	4.71
scotch	liquor	4.57
jazz	music	4.53
aircraft	plane	4.47
jurisdiction	law	4.45
summer	winter	4.38
⋮	⋮	⋮
texas	death	1.53
africa	theater	1.50
pennsylvania	writer	1.46
germany	worst	1.44
concrete	wings	1.43
recreation	dish	1.40

Table 7: Selected examples from top 10 and bottom 10 of MTurk-287 dataset

3.3.5. MTurk-771

MTurk-771 dataset[3] released by Halawi et al. (2012) contains 771 word pairs that were constructed using Amazon Mechanical Turk (AMT) to get the mean human-assigned relatedness judgements. The scores are on a scale from 1 to 5, where 1 stands for not related and 5 stands for highly related. Table 8 contains selected examples from top 10 and bottom 10 of this dataset according to the human judgements.

Word 1	**Word 2**	**Score**
female	woman	4.96
film	movie	4.91

[3] `http://www2.mta.ac.il/~gideon/mturk771.html`

quiet	silence	4.91
child	kid	4.86
ass	donkey	4.85
sight	vision	4.82
⋮	⋮	⋮
coat	newspaper	1.09
scandal	week	1.09
cup	son	1.09
beach	chain	1.05
shirt	tiger	1.042
afternoon	substance	1.0

Table 8: Selected examples from top 10 and bottom 10 of MTurk-771 dataset

3.3.6. MEN-3000

The MEN-3000 Test Collection[4] released by Bruni et al. (2014) contains 3000 English word pairs in along with the human-assigned similarity judgements, obtained by crowd-sourcing using AMT. The word pairs were randomly select from word that occur in ukWaC and Wackypedia combined[5] at least 700 times and in the open-sourced subset of the ESP game dataset[6] at least 50 times. The data collection for this dataset differs to WordSimilarity-353, since the annotators were asked to make binary decisions on which of two pairs are more related. The human-assigned similarity judgements is an integer between 0 and 50 due to the way the data was collected. Bruni et al. (2014) sampled the pairs in a balanced range of a text-based semantic score to avoid choosing unrelated pairs. The subjects of the study were not informed about the differences between similarity and relatedness and were presented with examples of similarity as a special case of relatedness. However, when describing the dataset Bruni et al. (2014) use both similarity and relatedness terms. Table 9 contains selected examples from top 10 and bottom 10 of this dataset according to the human judgements.

Word 1	**Word 2**	**Score**
sun	sunlight	50.0
automobile	car	50.0

[4] `https://staff.fnwi.uva.nl/e.bruni/MEN`

[5] `http://wacky.sslmit.unibo.it/doku.php`

[6] `http://www.cs.cmu.edu/~biglou/resources/`

river	water	49.0
stair	staircase	49.0
morning	sunrise	49.0
rain	storm	49.0
⋮	⋮	⋮
giraffe	harbor	1.0
feather	truck	1.0
festival	whisker	1.0
muscle	tulip	1.0
bikini	pizza	1.0
bakery	zebra	0.0

Table 9: Selected examples from top 10 and bottom 10 of MEN-3000 dataset

3.3.7. SimLex-999

SimLex-999 dataset[7] released by Hill et al. (2015) contains 999 word pairs and is used to evaluate computation models that learn meanings of words and concepts. There were multiple reasons that motivated the authors to create this dataset as supposed to using existing ones. It is common in NLP to have a performance upper bound on evaluation that is based on the average human performance or inter-annotator agreement (Resnik and Lin, 2010). One of the main reasons was that state-of-the-art models achieved the average performance of a human annotator on RG-65, WordSimilarity-353, MEN, and other gold standard datasets, which implies that the problem of similarity model has been resolved. Such implication is not true based on the performance of those models in automatically generated dictionaries, thesauri, or ontologies as observed by Hill et al. (2015). The authors argue that there are two further limitations in WordSimilarity-353 and MEN-3000 datasets The first limitation states that there is a high rating for many dissimilar word pairs. This phenomena can be observed in the previous section , where related word pairs ‘closet’ and ‘clothes’ seen in Tables 6 achieves a score that is higher than similar word pairs ‘train’ and ‘car’ seen in Tables 5. If WordSimilarity-353 rating for ‘closet’ and ‘clothes’ is 8, then SimLex-99 rating is 1.96 for the same pair. The second limitation described by Hill et al. (2015) is that word pairs pairs that are associated but not similar receive high ratings. RG dataset and subsequently the MC-30 dataset both are affected by the second limitation.

[7] `https://fh295.github.io/simlex.html`

In comparison to MEN-3000 or WordSimilarity-535 datasets, SimLex-999 provides a way to quantify the similarity between word pairs rather than association or relatedness. A computational model must therefore learn similarity of word pairs independent of association. This presents a challenge to the research community as most language-based models identify a relation between two words in terms of relatedness and conceptual association, since the relation is inferred based on their co-occurrence in corpora.

SimLex-999 dataset consists of 666 Noun-Noun pairs, 222 Verb-Verb pairs and 111 Adjective-Adjective pairs that are on a scale of 0 to 10. Furthermore, this dataset provides an independent concreteness score for a pair of words that provides how concrete word 1 and word 2 are conceptually. The intuition behind this score is that if broken down by part-of-speech, adjectives are more abstract than verbs which in turn are more abstract than nouns. Table 10 contains selected examples from top 10 and bottom 10 of this dataset according to the human judgements.

Word 1	**Word 2**	**Score**
vanish	disappear	9.80
quick	rapid	9.70
creator	maker	9.62
stupid	dumb	9.58
insane	crazy	9.57
large	big	9.55
⋮	⋮	⋮
island	task	0.30
gun	fur	0.30
chapter	tail	0.30
dirty	narrow	0.30
new	ancient	0.23
shrink	grow	0.23

Table 10: Selected examples from top 10 and bottom 10 of SimLex-999 dataset

3.3.8. YP-130

Yang and Powers (2006) released YP-130 dataset, which contains 130 word pairs, and measures semantic relatedness scores based on human judgements for verbs. Yang and

Powers (2006) identified 130 verb synonym tests from TOEFL[8] and ESL[9] language tests that assess the level of English for a non-native speaker for the university entry or employment. Human annotators were asked to indicate how strong the word pairs are related in meaning on an integer scale: not at all related (0), vaguely related (1), indirectly related (2), strongly related (3) and inseparably related (4). Table 11 contains selected examples from top 10 and bottom 10 of this dataset according to the human judgements.

Word 1	**Word 2**	**Score**
brag	boast	4.00
concoct	devise	4.00
divide	split	4.00
build	construct	4.00
end	terminate	4.00
accentuate	highlight	4.00
⋮	⋮	⋮
empty	situate	0.17
flush	spin	0.17
shake	swell	0.17
imitate	highlight	0.17
correlate	levy	0.00
refer	lean	0.00

Table 11: Selected examples from top 10 and bottom 10 of YP-130 dataset

3.3.9. Verb-143

The Verb-143[10] dataset released by Baker et al. (2014) contains 143 pairs of verbs along with the human judgement scores following the WordSimilarity-353 guidelines. 143 pairs of verbs were constructed from 122 unique verb lemma types, where each verb appears at least 10 times in total in the labour legislation and the environment datasets (Baker et al., 2014). For each word pair, an averaged human-annotator similarity score between 1 and 10 was assigned. Table 12 contains selected examples from top 10 and bottom 10 of this dataset according to the human judgements. It can be seen from that 4 word pairs in top 10 of most similar/related words share the same stem and are in a different tense.

[8] `https://www.ets.org/toefl`

[9] `https://www.esl-languages.com/en/study-abroad/adults/online-tests/index.html`

[10] `https://ie.technion.ac.il/~roiri/#data`

Word 1	**Word 2**	**Score**
refuses	refused	0.790
working	worked	0.780
seemed	seems	0.730
makes	produced	0.720
showing	showed	0.700
making	establishing	0.590
⋮	⋮	⋮
seemed	protects	0.100
refusing	exist	0.090
dismiss	finding	0.090
reducing	increased	0.080
produce	dismiss	0.070
starts	refused	0.070

Table 12: Selected examples from top 10 and bottom 10 of Verb-143 dataset

3.3.10. SimVerb-3500

The SimVerb-3500[11] dataset released by Gerz et al. (2016) contains 3500 verb pairs with semantic similarity ratings on a scale from 0 to 10, where 0 means not similar at all and 10 means synonymous. One of the main motivations for creating this dataset was the limited amount of data in the previous gold datasets such as Verb-143, YP-130 and a subset of SimLex-999 when evaluating verb similarity. When creating SimLex-999 dataset Hill et al. (2015) provided guidelines as to what constitutes to be a high-quality evaluation resource, where three criteria were provided: representative, clearly defined and consistent. When constructing SimVerb-3500 Gerz et al. (2016) followed the same annotation guidelines as for SimLex-999 to satisfy all criteria. SimVerb-3500 dataset contains 827 verb types, all normed verb types from the USF free-association database[12], and provides 3 member verbs for each top-level VerbNet[13] class. These two standard semantic resources, therefore, provided a wide coverage of verb pairs, ensuring the first criteria of representation if fulfilled. Furthermore, the annotators were explicitly instructed to give low ratings to antonyms and to make a distinguish between similarity and relatedness, which covers the limitations of the previously

[11] `http://people.ds.cam.ac.uk/dsg40/simverb.html`
[12] `http://w3.usf.edu/FreeAssociation/`
[13] `http://verbs.colorado.edu/verb-index/`

existing benchmarks explained by Hill et al. (2015). Table 13 contains selected examples from top 10 and bottom 10 of this dataset according to the human judgements.

Word 1	**Word 2**	**Score**
repair	fix	9.96
rip	tear	9.96
build	construct	9.96
flee	escape	9.79
triumph	succeed	9.79
obtain	acquire	9.79
⋮	⋮	⋮
go	stay	0.00
shut	vomit	0.00
accept	decline	0.00
create	dive	0.00
lose	keep	0.00
freeze	thaw	0.00

Table 13: Selected examples from top 10 and bottom 10 of SimVerb-3500 dataset

3.3.11. RW

The Rare-Words (RW)[14] dataset released by Luong et al. (2013) focused on constructing a dataset on rare words to complement existing datasets on the frequent words. In order to construct the data Luong et al. (2013) first selected a list of rare words, then found a word (not necessarily rare) to form a word pair, and finally collected a human judgements score on how similar each pair is. The selection of rare words was done by sampling words from various frequency bins and the affixes they possess. To prevent selection of a non-English word, one of the requirements for a word to be sampled was that it had a non-zero number of synsets in WordNet (Miller (1995)). RW contains 2034 word pairs at human similarity ratings on a scale from 0 to 10 collected by AMT. Due to the nature of the dataset, the human annotators were asked to indicate if they are familiar with first word, second word or neither, hence discarding pairs that had less than a particular number of scores. Table 14 contains selected examples from top 10 and bottom 10 of this dataset according to the human judgements.

[14] `https://nlp.stanford.edu/~lmthang/morphoNLM/`

Word 1	**Word 2**	**Score**
decapitated	headless	10.00
cheapen	devalue	10.00
nonnative	foreign	10.00
symmetrical	balanced	10.00
conjecture	hypothesis	10.00
liveable	habitable	10.00
⋮	⋮	⋮
intertwining	raw	0.00
recorders	box	0.00
grinder	wisdom	0.00
stockers	animal	0.00
characters	scratch	0.00
prospector	sourdough	0.00

Table 14: Selected examples from top 10 and bottom 10 of RW dataset

3.4. Evaluation Metrics

Before the emergence of WordSimilarity-353 dataset described in Section 3.3.2 it was a common practice among researchers to perform evaluation with Pearson correlation often denoted as r. Agirre et al. (2009) argue that one of the drawbacks of using Pearson correlation is that this metric is less informative when the scores of two variables are not linearly correlated, since Pearson correlation asses linear relationships. As such, Agirre et al. (2009) proposed using Spearman correlation often denoted as ρ , which is independent of the dataset and can assess the strength and direction of monotonic relationships. Hence, in order to evaluate performance on each of the datasets described in Section 3.3, Spearmans rank correlation coefficient between two variables can be used, where the first variable is the human judgement score and the second variable are scores produced by the computational model. Spearmans rank correlation coefficient is a score between -1 and 1 and indicates the direction of association between the human judgements (independent variable) and scores produced by a computation model (dependent variable). If the model predictions increases when the human judgement score increases, then ρ is positive, else if model predictions decreases when human judgement score increases, then ρ is negative. If the model predictions does not increase or decrease when human judgement score increase, the value of ρ is 0.

ρ is equal to 1 when human judgement score and model score are perfectly monotonically related.

3.5. Approach

In order to perform evaluation for each word in all datasets we extracted up to 100 corresponding image-based features explained in Section 2.2. Table 15 displays the dataset, the number of word pairs for each of the corresponding dataset and the number of words missing in the data collection of Callahan (2017). In total there are 5846 unique words and 11109 word pairs. For these 5864 words the average of the maximum number of images is 94.4. We say a word pair is missing in the data collection if either the first word or the second word is not present. Other than for RW dataset, the coverage for word pairs is sufficient to perform a set of experiments. For the retrieved words we extracted a linguistic representation and used word2vec (Mikolov et al., 2013) trained on Google News 100B[15].

Dataset	**Number of Word Pairs**	**Number of Word Pairs Missing**
RG-65	65	0
MC-30	30	0
WordSimilarity-353-ALL	353	0
WordSimilarity-353-SIM	203	0
WordSimilarity-353-REL	252	0
MTurk-287	287	5
MTurk-771	771	1
MEN-3000	3000	1
SimLex-999	999	1
YP-130	130	0
VERB-143	144	0
SimVerb-3500	3500	17
RW	2034	711

Table 15: Number of word pairs that are missing from image-based feature for a given dataset names and their corresponding number of word pairs

We extended the existing scripts[16] for evaluating word vectors released by Faruqui and Dyer (2014). The original script provides an implementation of Spearman correlation rank

[15] http://magnitude.plasticity.ai/word2vec/medium/GoogleNews-vectors-negative300.magnitude

[16] https://github.com/mfaruqui/eval-word-vectors

coefficient and a way to compute it given an input dataset and an input file containing word to vector mapping. For our experiments, we extended the script to support an input file containing a value mapping to a word pair in the input dataset, and an ability to read the first n dimensions of a vector if an original input file is specified.

3.6. Experiments and Results

The following subsections describe the performance of various approaches taken to generate visual representation.

3.6.1. Avg(w1, w2)

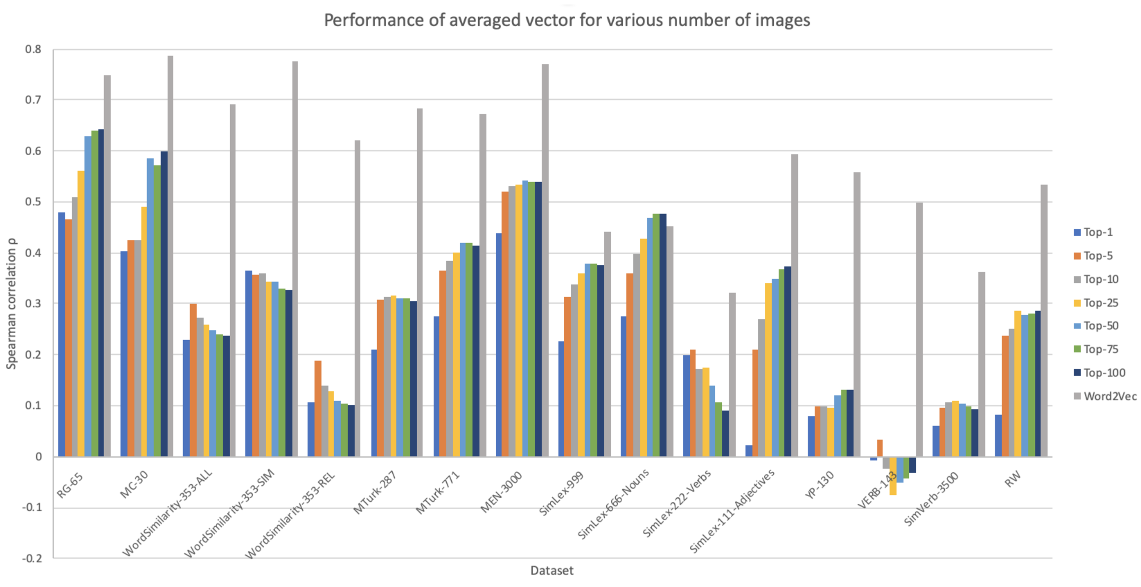


Figure 10: Performance of the averaged vector for various number of images on all datasets. (Data taken from Table 31)

Figure 10 displays the performance of the averaged vector Avg(w1, w2) for various number of images (1, 5, 10, 25, 50, 75, 100) on all datasets described in Section 3.3. In order to get a similarity score for a pair of words, the cosine similarity is taken between the two averaged vectors. As a reference point, word2vec performance in grey is plotted in addition to visual representations. Based on the Figure, it can be seen that as the number of image increases the performance increases for almost all of the datasets other than WordSimilarity-353-ALL, SimLex-222-Verbs, Verb-143, and SimVerb-3500. An important observation is that the performance of visual embeddings is comparable to the performance on SimLex-999 dataset, which is thought to overcome limitations of datasets like WordSimilarity-353 and MEN-3000 described in Section 3.3.

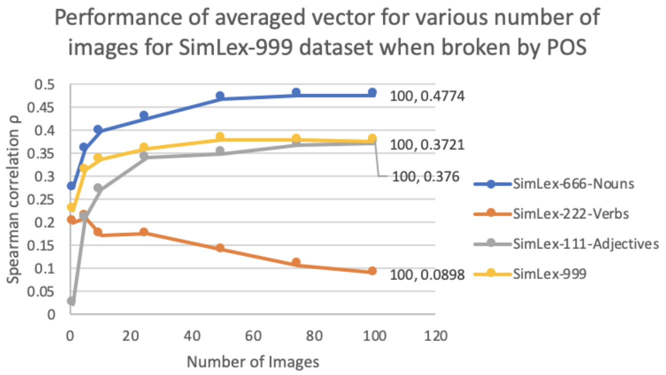


Figure 11: Performance of the averaged vector on SimLex-999 dataset for various number of images and POS. (Data taken from Table 31)

Figure 11 displays the performance of the averaged vector for various number of images for the SimLex-999 dataset in particular with the breakdown by POS. It can be seen that the overall performance on the SimLex-999 dataset increases as the number of images increases, since the performance increases for 666 nouns and 111 adjectives in this dataset. However, the performance significantly decreases for 222 verbs when the number of images increases. In addition, curves in the Figure suggest there is a plateau between 50 and 75 images for the performance for 666 nouns and 111 adjectives.

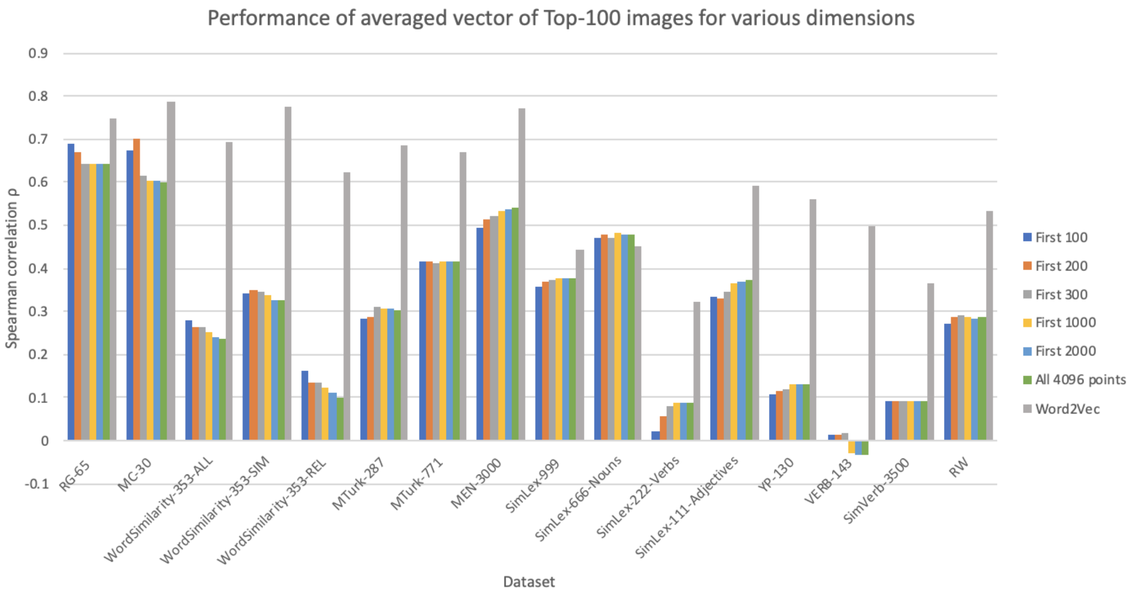


Figure 12: Performance of the averaged vector for Top-100 images on all datasets for various number of dimensions (Data taken from Table 34)

Figure 12 displays the performance of averaged vector for the top 100 images on all datasets with varying number of dimensions (100, 200, 300, 1000, 2000, 4096). The idea behind such experiment is to get an idea whether simply taking first n number of points is results into a drop in performance in comparison to taking all 4096 features. Based on this Figure, there is an improvement in performance for almost all datasets excluding RG-65, MC-30, WordSimilarity-353-ALL. This decrease in performance can be explained by the fact that MC-30 is a subset of RG-65, which in turn is a subset of WordSimilarity-353-ALL. Perhaps the small size of the dataset resulted in a slight decrease of a performance as the number of dimensions increase, which is especially evident for related words in WordSimilarity-353-REL. Overall, the results based on this set of experiments are counter-intuitive, since taking the first 300 points of 4096 yields the performance that is not substantially different than taking all data points.

3.6.2. AvgMax

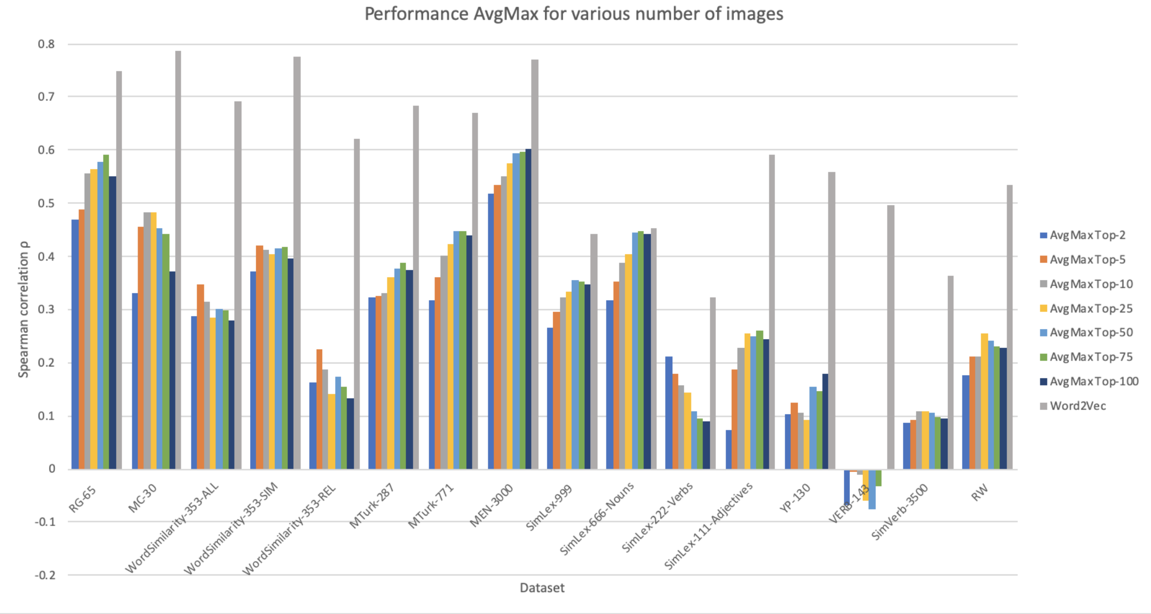


Figure 13: Performance of AvgMax for various number of images on all datasets. (Data taken from Table 32)

Figure 13 displays the performance of the AvgMax for various number of images (2, 5, 10, 25, 50, 75, 100) on all datasets described in Section 3.3. In order to get a similarity score for a pair of words, the scalar value is produced as a result of performing AvgMax on a specified number of images. Similarly to Avg(w1, w2) in Figure 10, the performance of visual embeddings increases as the number of images increases for datasets other than WordSimilarity-353-ALL, SimLex-222-Verbs, Verb-143, and SimVerb-3500. There are subtle differences in performance between Figure 10 and Figure 13, however the overall trend stays the same.

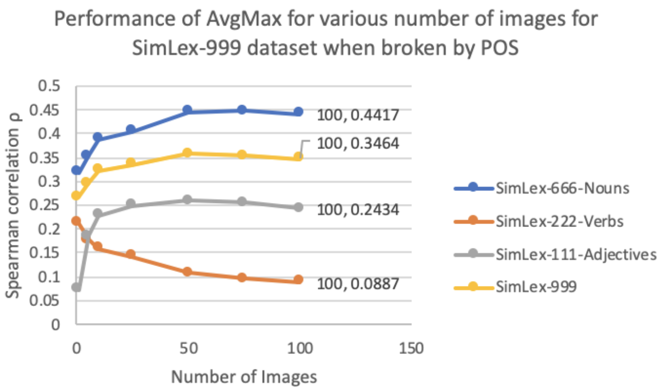


Figure 14: Performance of AvgMax on SimLex-999 dataset for various number of images and POS. (Data taken from Table 32)

Figure 14 displays the performance of AvgMax for various number of images for the SimLex-999 dataset broken down by POS. Similarly to Figure 11, in this Figure the overall performance on the SimLex-999 dataset increases as the number of images increases, however for 222 verbs the performance significantly decreases. If Figure 14 and Figure 11 are to be compared, it seems that for 111 adjectives the performance of AvgMax slightly degrades as the number of images reaches 50, which results in a lower overall performance than in Avg(w1, w2) .

3.6.3. AvgAvg

Similarly to the previous subsection, Figure 15 displays the performance of the AvgAvg for various number of images on all datasets described. The results are very similar to results for AvgMax described in Figure 13. What can be noted here is that the performance of AvgAvg on SimLex-999 dataset is by slightly lower (0.1) than AvgMax.

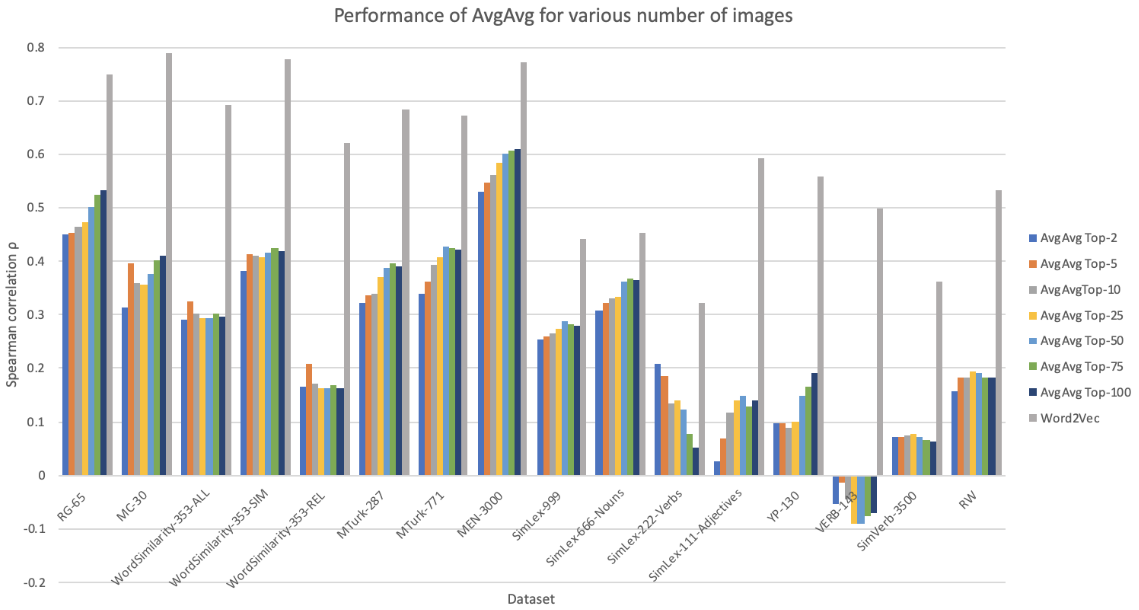


Figure 15: Performance of AvgAvg for various number of images on all datasets. (Data taken from Table 33)

3.6.4. Comparison of models

A few other approaches were attempted before comparing the performance for various representations. One of the approaches was to perform dimensionality reduction via Principal Component Analysis (PCA) on the vectors prior to producing an averaged vector. The motivation behind applying dimensionality reduction on 4096 features was that it would allow to perform concatenation of word2vec features and the reduced vector to combine two unimodal representations into a multimodal representation. The reason why PCA is needed to be applied is because the result of concatenating the original 4096-dimensional vector with the 300-dimensional word2vec as the means of producing multimodal representation had a similar performance as using 4096-dimensional vector by itself without concatenation. We experimented the reduction to 300 dimensions, the same dimension as word2vec, as it seemed fair to use the input to two modes of data evenly.

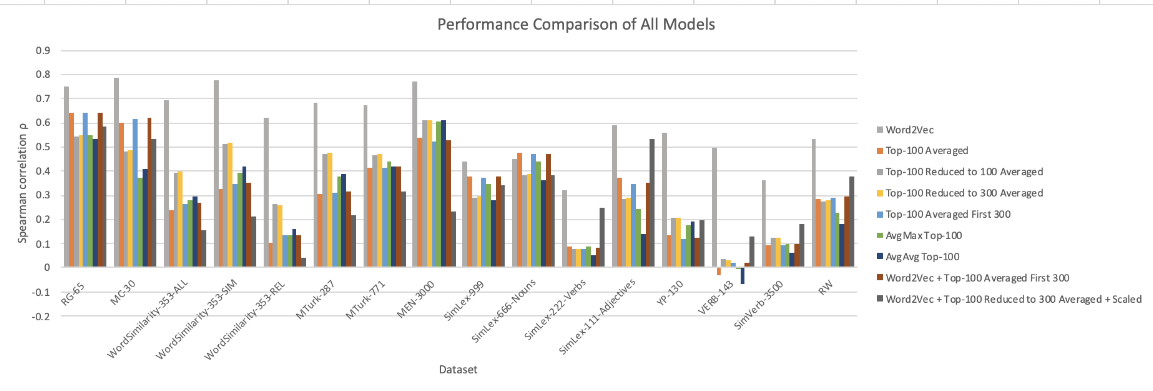


Figure 16: Performance comparison of various models. (Data taken from Tables 31, 32, 33, 34 and 35)

Figure 16 displays the performance of various models on the given datasets. Based on the results presented in the Figure it can be seen that linguistic word2vec representation still surpasses the visual representation on all thirteen datasets. Moreover, concatenation of word2vec with the visual representation does not yield better results as using word2vec by itself. Such results have been observed by the previous work of Wang et al. (2018).

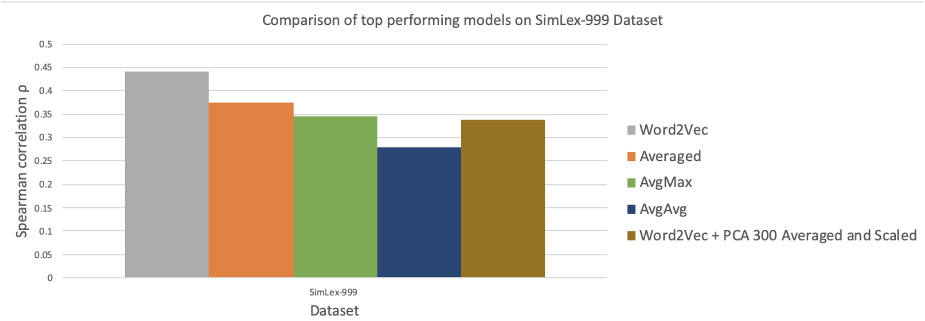
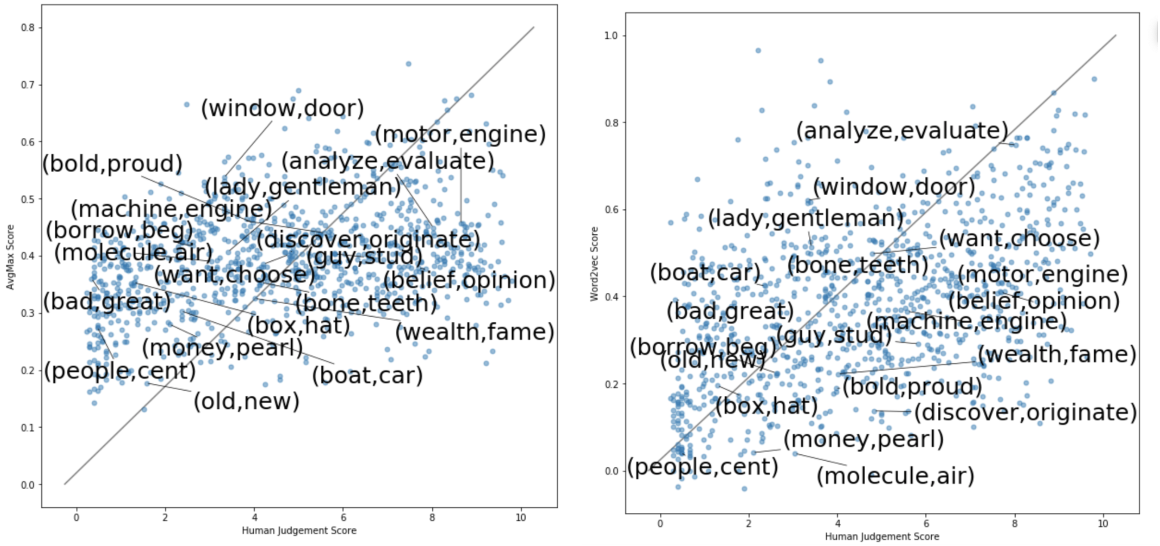


Figure 17: Performance comparison of selected models on SimLex-999 dataset. (Data taken from Tables 31, 32, 33, 34 and 35)

Due to the different origin, size, and nature of the datasets, it is perhaps better to compare the performance of models against one dataset, and in particular, SimLex-999 due to reasons mentioned in Section 3.3. Figure 17 presents the comparison of selected top performing models on SimLex-999 dataset. As can be seen from the Figure, the performance of linguistic word2vec representation is slightly better but comparable to the performance of visual and multimodal representations. It can be seen that Avg(w1, w2) and Avg-Max achieve best performances out of the selected visual representations followed by a multimodal representation of word2vec and PCA to 300 of Avg(w1, w2).

3.6.5. *Qualitative Analysis*



(a) Scatter-plot of AvgMax score vs Human Judgement Score

(b) Scatter-plot word2vec score vs Human Judgement Score

Figure 18: AvgMax and word2vec side-by-side comparison of selected word pairs in the SimLex-999 dataset

Figure 18 provides a side-by-side comparison of AvgMax and word2vec for selected word pairs from the SimLex-999 dataset. The x-axis in both of those plots is the human judgement score on the scale from 0 to 10. The y-axis in Figure 18a is the predicted score by AvgMax, the y-axis in Figure 18b is the predicted score of word2vec, both of which are on a scale from 0 to 1, since both use cosine similarity between vectors. The better the representation, the closer data points are to the diagonal. It can be seen from the Figures that word2vec has data points more in a more sparse range, whereas AvgMax's range is more dense. This is an interesting observation, and further analysis needs to be made to understand the reason for AvgMax's scores to be in a smaller range. It seems that both models have a lot in common in terms of predicting, for instance the word pair ‘bad, great’ for both predictions appears to be relatively close to the diagonal, however the value that prediction assigns is greater than the human judgement score.

Word 1	**Word 2**	**Human Score**	**AvgMax Score**	**word2vec Score**
vanish	disappear	9.80	0.356	0.9
quick	rapid	9.70	0.322	0.498
creator	maker	9.62	0.365	0.261
stupid	dumb	9.58	0.541	0.817
insane	crazy	9.57	0.425	0.734
large	big	9.55	0.314	0.556
⋮	⋮	⋮	⋮	⋮
island	task	0.30	0.241	0.035
gun	fur	0.30	0.162	0.139
chapter	tail	0.30	0.31	0.101
dirty	narrow	0.30	0.319	0.113
new	ancient	0.23	0.268	0.166
shrink	grow	0.23	0.335	0.571

Table 16: Selected examples from top 10 and bottom 10 of SimLex-999 dataset and the predicted AvgMax and word2vec scores

Table 16 contains 6 examples of most similar word pairs and 6 least similar word pairs from the SimLex-999 dataset along with the predictions made by AvgMax and word2vec. It can be seen from the AvgMax scores that the range of values for least similar and most similar word pairs is almost negligible, which is consistent with the observation in Figure 18. From Table 16 it can also be observed that AvgMax and word2vec scores predict similar scores, but the range for AvgMax is smaller. For instance, they both predict for the word pair ‘shrink, grow’ relatively high scores even though in SimLex-999 dataset this word pair is considered to be the least similar. Same pattern can be observed for one of the most similar pairs such as ‘creator, maker’, since both AvgMax and word2vec assign a relatively low score.

CHAPTER 4 : Clustering Paraphrases By Word Sense

4.1. Overview

For this part of the research, we seek to understand how to cluster paraphrases by word sense using visual-based word representations or a combination of visual and linguistic representations. This chapter contains previous work, descriptions of the datasets, generation of a new dataset, along with the experiments and results.

4.2. Literature Review

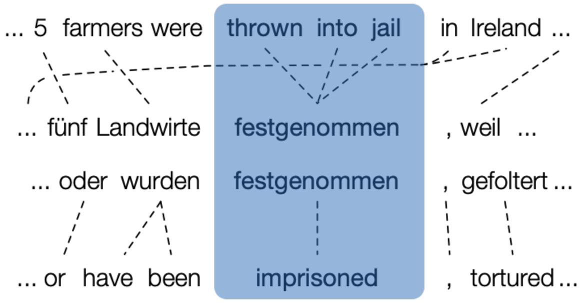

... 5 farmers were thrown into jail in Ireland ...
... fünf Landwirte festgenommen , weil ...
... oder wurden festgenommen , gefoltert ...
... or have been imprisoned , tortured ...

Figure 19: A bilingual pivoting method assumes two strings have the same meaning if they translate to the same foreign string. The method then pivots over bilingual parallel corpus to extract paraphrases

Paraphrases are different textual representations that maintain the same meaning. The Paraphrase Database (PPDB) contains over 100 million paraphrases in 23 languages generated using the bilingual pivoting method (Bannard and Callison-Burch, 2005), which posits that two words are potential paraphrases of each other if they share one or more foreign translations. Figure 19 displays the bilingual pivoting method that finds a pair of English words ‘thrown into jail’ and ‘imprisoned’ to be paraphrases, since both translate to a German word ‘festgenommen’. The paraphrases in PPDB are already partitioned by syntactic type, following the work of Callison-Burch (2008). This is to say that the paraphrases of the noun representation of a word would be separated from its verb representation. However, there is still the inherent problem of dealing with paraphrases of different senses within the same syntactic type.

In earlier work by Apidianaki et al. (2014) a graph-based method was developed to cluster the paraphrases of PPDB by word sense. In this approach, the paraphrases are represented as nodes and pairs of words which share one or more foreign alignments are linked, with the edge weighted by the contextual similarity between the two words. The clusters are computed by removing the edges with similarity values below certain threshold and extracting the remaining connected components, which are the final sense clusters.

We follow a similar approach to the research done by by Cocos and Callison-Burch (2016) that explored more advanced clustering algorithms and similarity measures. A key component to clustering is to build a similarity or affinity matrix that would represent a pairwise similarity between two paraphrases. Cocos and Callison-Burch (2016) experimented with several text-based measures of affinity between two paraphrases, such as second-order paraphrases (Pavlick et al., 2015) and distributional semantics. Their work yielded sense clusters which were qualitatively as well as quantitatively good.

4.3. Methodology

4.3.1. Graph Clustering

To create clusters, we use the Self-Tuning Spectral Clustering algorithm Zelnik-Manor and Perona (2004), which is an improved version of the Spectral clustering that creates a flat clustering for a pre-specified number of clusters. Cocos and Callison-Burch (2016) found that Self-Tuning Spectral Clustering had one of the best performances under different similarity matrices. Self-Tuning Spectral Clustering algorithm projects data into a lower dimensional space where it is more easily separable. In our experiments we denote this clustering algorithm simply as Spectral.

Most clustering algorithms, including Spectral clustering, require two inputs:

1. a number of clusters k.

2. an adjacency / similarity / affinity matrix A, a non-negative symmetric matrix where the similarity between words w_i and w_j is stored in cell a_{ij} visualised in Figure 20.

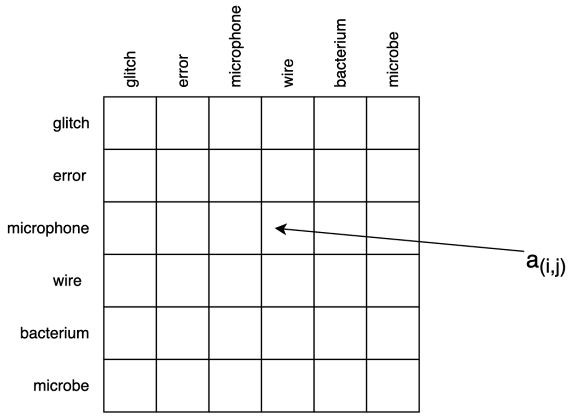


Figure 20: An adjacency / similarity / affinity matrix A, a non-negative matrix where the similarity between words w_i and w_j is stored in cell a_{ij}

In order to ge the number of clusters k, two approaches have been chosen. In the first approach the number of clusters k is known ahead of time based on the gold clusterings. In the second approach k is inferred by re-running the clustering with several possible values and choosing the clustering that has the highest mean Silhouette Coefficient (Rousseeuw, 1987). The silhouette score measures how similar a point is to its cluster and dissimilar to other clusters and can be seen in Equation 4.1. Silhouette coefficients in the context of the described clustering algorithm is $\text{SC} = 1 - A$, since this matrix denotes pairwise distances.

$$\text{SC} = \frac{b(p_i) - a(p_i)}{\max(a(p_i), b(p_i))} \tag{4.1}$$

where

$b(p_i)$ = lowest average distance from p_i to the nearest external centroid

$a(p_i)$ = average distance from p_i to each other p_j in the same cluster

The following subsections describe existing and novel similarity measures to populate or fill the similarity matrix A visualised in Figure 21.

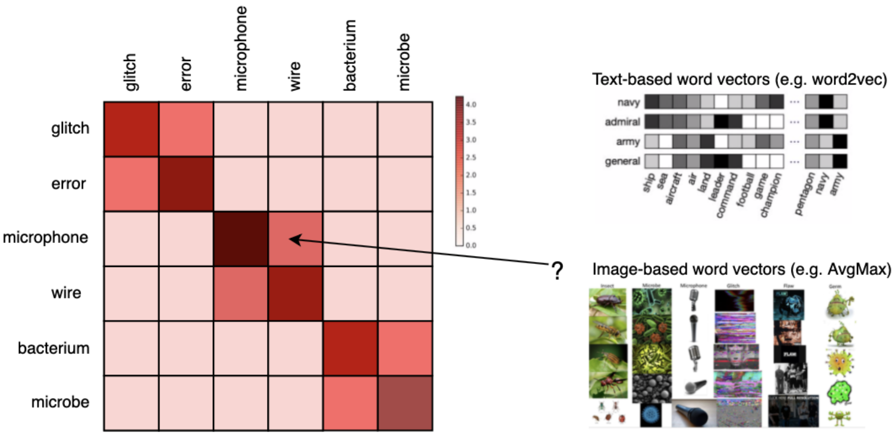


Figure 21: Approaches to populate similarity matrix A: text-based and image-based

4.3.2. Existing Similarity Measures

Cocos and Callison-Burch (2016) introduced and implemented 4 different similarity measures:

- Paraphrase Scores $PPDB_{2.0}Score$ (PPDB2)
- Second-order Paraphrase Scores $sim_{PPDB.cos}$ and $sim_{PPDB.js}$
- Similarity of Foreign Word Alignments sim_{TRANS}
- Monolingual Distributional Similarity $sim_{DISTRIB}$ (word2vec (Mikolov et al., 2013))

Based on their experimental results, the two best performing similarity measures for forming the similarity matrix and silhouette coefficients are $PPDB_{2.0}Score$ (PPDB2) and $sim_{DISTRIB}$ (word2vec (Mikolov et al., 2013)). PPDB2 scores (Pavlick et al., 2015) are non-negative real number between a pair of words that were judged by human annotators for the paraphrase quality of possible word pairs.

For the Spectral clustering method, the best configuration happens when PPDB2 scores are used for similarities and word2vec are used for silhouette coefficients. As such, we decided to use PPDB2 scores and word2vec similarity measures for our initial set of experiments.

4.3.3. New similarity measures

Images In this approach, we populate the similarity matrix using AvgMax of top 100 images described in Section 2.3.2. The motivation of using AvgMax as supposed to AvgAvg comes from performance results in Chapter 3.

Contextual Information In this approach, we populate the similarity matrix based on the contextual information of images used to compute image features. For each word in a paraphrase set of a target word, we retrieve up to 100 html pages that are linked to 100 images from the dataset. For a word in a paraphrase set, we join all of the collected html pages into a single document with a new line. This means that one paraphrase in a set maps to one document with all html pages joined. We then build a Term Frequency - Inverse Document Frequency (TF-IDF) using sub-linear TF Scaling for a paraphrase set, where the row maps to a vector representation of a paraphrase. We then populate a similarity matrix by taking the cosine similarity between the extracted embedding. What is important to note is that the vocabulary used to build TF-IDF matrix is not the overall vocabulary of the paraphrase file, but consists of vocabulary occurring inside the paraphrase set.

4.3.4. Combining similarity measures

As can be inferred, there are numerous ways to combine existing and new similarity measures to populate the similarity matrix. In our approach, we chose to simply average the scores produced by various representations and apply L2 norm to already populated similarity matrix. However, there are other proposed ways to combine the multiple modes of similarities as explained in Chapter 5.

4.3.5. Evaluation Measures

We use the same evaluation measures as was used by 2010 SemEval Word Sense Induction Task (Manandhar et al., 2010), Apidianaki et al. (2014) and Cocos and Callison-Burch (2016). The two evaluation measures are Paired F-Score and V-Measure. To compute paired F-Score all possible word pairs are labelled as being in the same cluster or not. The labelling of word pairs is done for both predicted and ground-truth or gold clusterings. The F-Score is then computed on the labelled word pairs using precision and recall. V-Measure is an entropy-based measure which explicitly measures how successfully the criteria of homogeneity and completeness have been satisfied. Homogeneity denotes conditional entropy of the class distribution given the clustering and completeness denotes the opposite. What is important to note is that these evaluation metrics are averages of paired F-Score and V-Measure for each polysemous word, weighted by the number of clusters in for this word in the gold file. In order to find the best clustering method performance, we seek to find balance between paired F-Score and V-Measure.

4.3.6. Incorporating entailment

PPDB2.0 (Pavlick et al., 2015) contains automatically predicted semantic entailment relationships such as equivalence, exclusive, independent, forward and reverse. Cocos and Callison-Burch (2016) exploit positive entailment relationship of equivalence and forward/reverse entailment by multiplying each pairwise entry by entailment probability and recording the result in the similarity matrix after it has been normalised.

4.3.7. Baselines

Similarly to Cocos and Callison-Burch (2016) we implemented the following baselines:

- Most Frequent Sense (MFS) = all paraphrases in a single cluster
- One Cluster per Paraphrase (1c1Par) = each paraphrase in a paraphrase list has its own cluster
- Random (RAND) = randomly assigns paraphrases to k clusters, where k is static and is equal to 5

The intuition behind these baselines is that 1c1Par favours V-Measure, whereas MFS favours F-Score, so the top performing model should take a position in between the two extremes.

The main challenge we faced during evaluation was the lack gold standard clusters against which we could compare our solutions. Therefore, we started evaluating our results against WordNet+ and CrowdClusters datasets used by Cocos and Callison-Burch (2016) and described in the subsequent sections.

4.4. Wordnet+ Dataset

4.4.1. Description

WordNet+[1] paraphrase file contains 201 polysemous words from the SEMEVAL 2007 dataset, where a list of paraphrases originates from the intersection of the PPDB 2.0 XXXL paraphrases, WordNet synsets and their immediate hyponyms and hypernyms. Gold clusterings[2] for this dataset consist of a WordNet synset along with the hypernyms and hyponyms of words for a given synset.

Tables 17, 18 and 19 provide paraphrase and gold files statistics for the WordNet+ dataset. Tables 17 and 18 display information, such as the number of paraphrases for a target word in both paraphrase and gold files. Table 19 provides information from the gold file about the number of clusters per target word as well as the number of paraphrases within the corresponding cluster.

POS	**No of target words**	**No of paraphrases**	**No of unique paraphrases**	**Mean No of paraphrases per target word**	**Median No of paraphrases per target word**	**Std No of paraphrases per target word**
noun	65	1222	1007	18.8	16	12.2
verb	56	2918	1271	52.1	45.5	40
adjective	59	424	344	7.2	7	4.3
adverb	35	224	169	6.4	6	2.5
all	215	4788	2953	22.3	11	28.4

Table 17: WordNet+ Paraphrase File POS Breakdown – Number of Paraphrases Statistics

[1] https://github.com/acocos/cluster_paraphrases/blob/master/data/pp/combined_semeval_handpicked_multiword_xxxl_PPDB2.0Score_plusself_wnfilt.ppsets

[2] https://github.com/acocos/cluster_paraphrases/blob/master/data/gold/wordnet_eval_targets.wngold

POS	No of target words	No of paraphrases	No of unique paraphrases	Mean No of paraphrases per target word	Median No of paraphrases per target word	Std No of paraphrases per target word
noun	60	4621	3915	77	73	45.4
verb	52	8565	4355	164.7	111.5	169.3
adjective	54	754	625	14	11	9.1
adverb	35	275	213	7.9	7	3.7
all	201	14215	8667	70.7	32	109.1

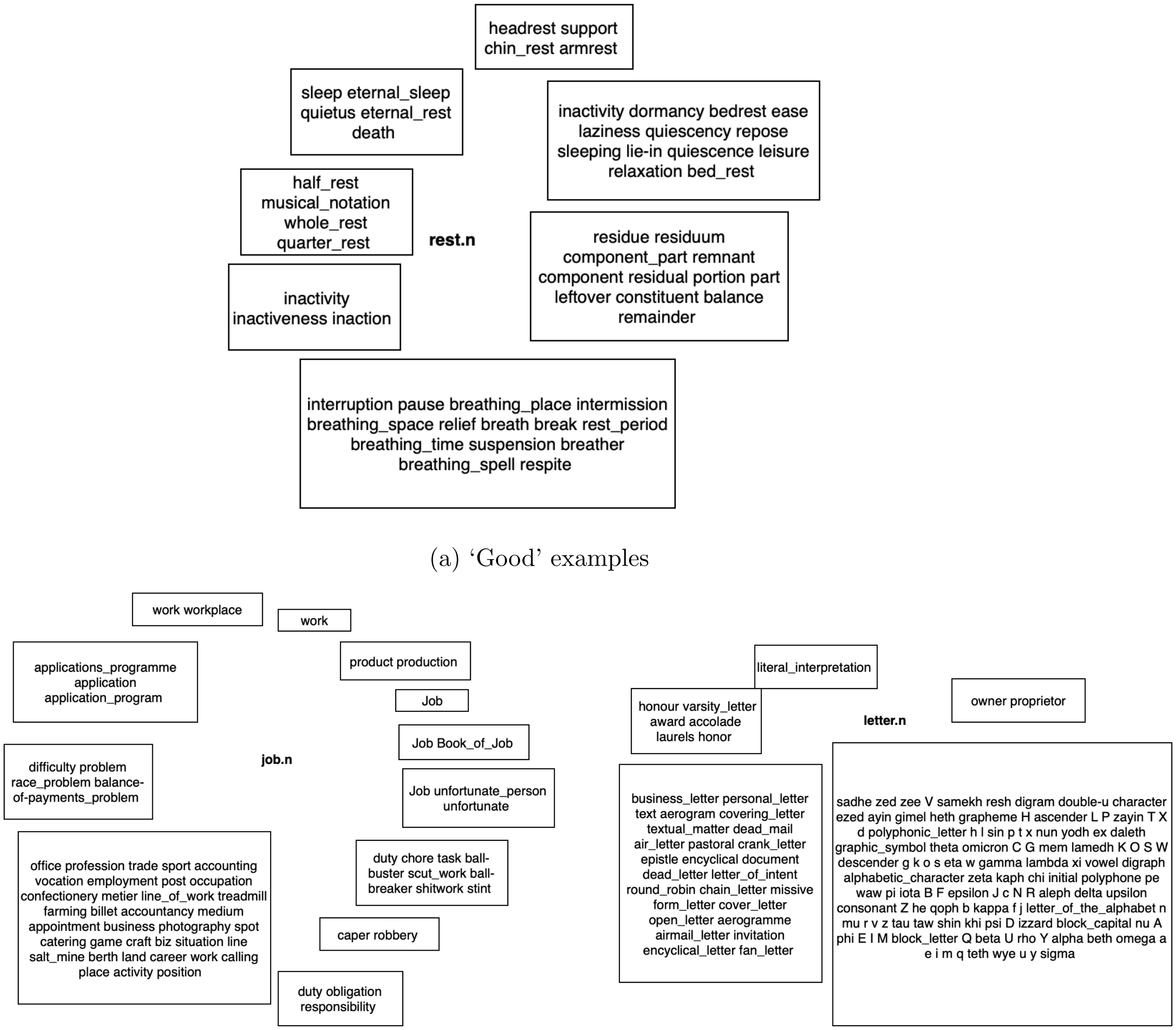
Table 18: WordNet+ Gold File POS Breakdown – Number of Paraphrases Statistics

POS	Mean No of clusters per target word	Median No of clusters per target word	Std No of clusters per target word	Mean No of paraphrases per cluster	Median No of paraphrases per cluster	Std No of paraphrases per cluster
noun	8.1	7	5	9.6	6	12.9
verb	16	11.5	11	10.3	5	29.3
adjective	5.5	4.5	3.2	2.6	2	2.4
adverb	3.1	3	1.2	2.5	2	1.9
all	8.5	6	8	8.3	4	21.8

Table 19: WordNet+ Gold File POS Breakdown – Number of Cluster and Paraphrases within a Cluster Information

A few details can be observed from this evaluation of the paraphrase and gold files. To start with, there are three times more paraphrases in the gold file as there are in the paraphrase file. As a consequence, it can be seen in Tables 17 and 18 that there is a difference in proportion between the mean, median and standard deviation in number of paraphrases for a target word for nouns and verbs. According to Table 19 the number of clusters per target word is around 8.5 across all words, however, the mean and median number of clusters per each POS varies, in particular, with adverbs having the least number of clusters and verbs have the most number of clusters. The mean number of clusters per target word for adverbs can be justified by the little amount of data in comparison to other POS. Based on these observations, it can be concluded that there is a need to separate the evaluation by POS, as the number of paraphrases supplied as an input as well as the number of clusters expected as an output for a target word varies between different types of words for WordNet+ dataset.

4.4.2. Gold Clusters Examples


headrest support chin_rest armrest
sleep eternal_sleep quietus eternal_rest death
inactivity dormancy bedrest ease laziness quiescency repose sleeping lie-in quiescence leisure relaxation bed_rest
half_rest musical_notation whole_rest quarter_rest
rest.n
residue residuum component_part remnant component residual portion part leftover constituent balance remainder
inactivity inactiveness inaction
interruption pause breathing_place intermission breathing_space relief breath break rest_period breathing_time suspension breather breathing_spell respite
work workplace
work
applications_programme application application_program
product production
Job
Job Book_of_Job
difficulty problem race_problem balance-of-payments_problem
job.n
Job unfortunate_person unfortunate
office profession trade sport accounting vocation employment post occupation confectionery metier line_of_work treadmill farming billet accountancy medium appointment business photography spot catering game craft biz situation line salt_mine berth land career work calling place activity position
duty chore task ball-buster scut_work ball-breaker shitwork stint
caper robbery
duty obligation responsibility
literal_interpretation
honour varsity_letter award accolade laurels honor
owner proprietor
letter.n
business_letter personal_letter text aerogram covering_letter textual_matter dead_mail air_letter pastoral crank_letter epistle encyclical document dead_letter letter_of_intent round_robin chain_letter missive form_letter cover_letter open_letter aerogramme airmail_letter invitation encyclical_letter fan_letter
sadhe zed zee V samekh resh digram double-u character ezed ayin gimel heth grapheme H ascender L P zayin T X d polyphonic_letter h l sin p t x nun yodh ex daleth graphic_symbol theta omicron C G mem lamedh K O S W descender g k o s eta w gamma lambda xi vowel digraph alphabetic_character zeta kaph chi initial polyphone pe waw pi iota B F epsilon J c N R aleph delta upsilon consonant Z he qoph b kappa f j letter_of_the_alphabet n mu r v z tau taw shin khi psi D izzard block_capital nu A phi E I M block_letter Q beta U rho Y alpha beth omega a e i m q teth wye u y sigma

(a) 'Good' examples

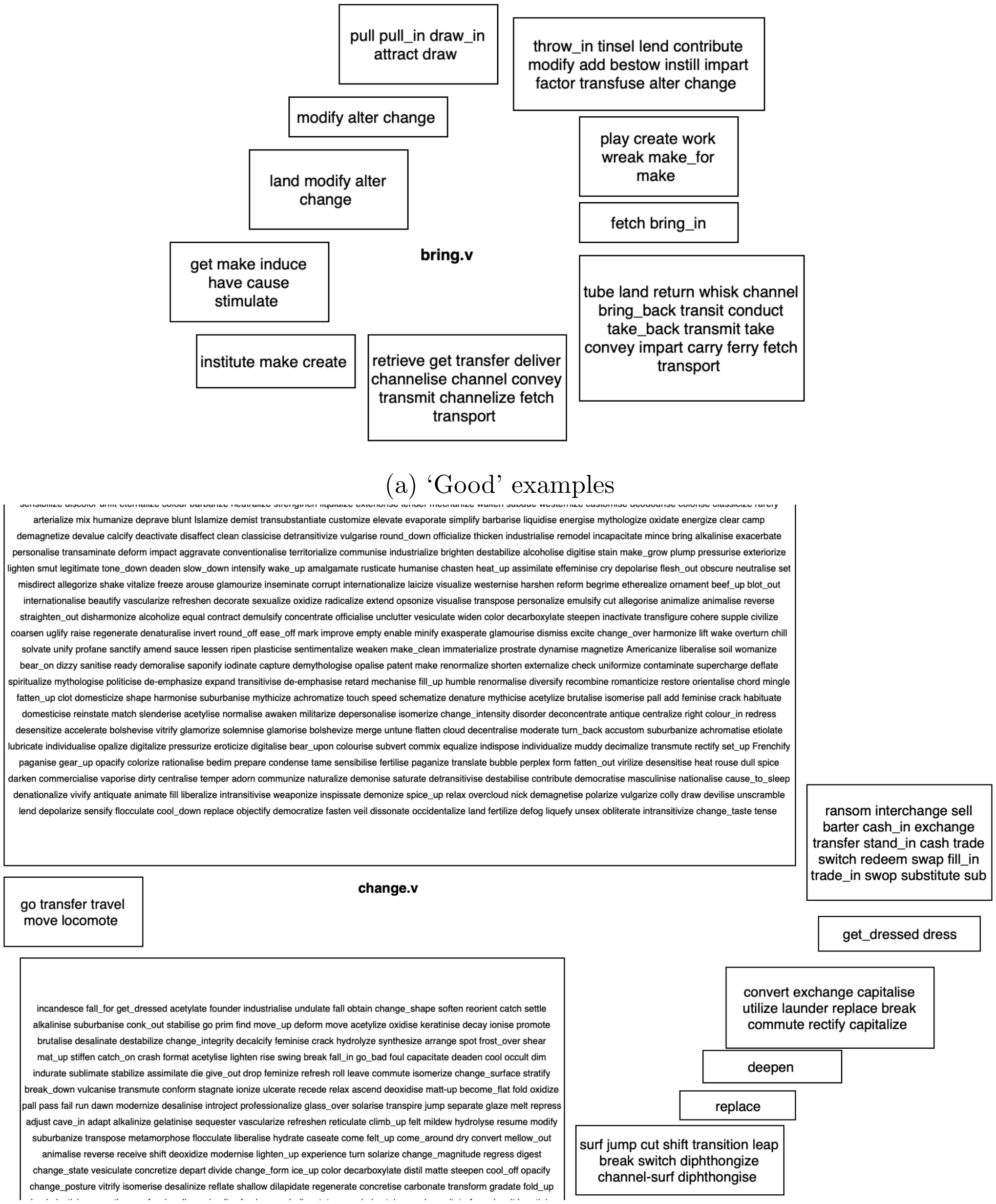
(b) 'Bad' examples

Figure 22: Examples for nouns from the WordNet+ Gold File

Nouns Figure 22 contains selected examples of clusterings for nouns from the WordNet+ gold file. Based on our observations, the clustering for the word 'rest.n' seen in Figure 22a contains a good representation of distinct senses. For 'rest.n' a few senses can be clearly observed: sleep, inactivity, pause, musical notation, and component remainder from mathematics. Figure 22b contains gold clustering for the word ‘job.n’ and ‘letter.n’. The reason we believe ‘job.n’ does not separate paraphrases in a clean manner is because there are many clusters that are very specific and obfuscate the more general senses of the target word. For instance the clusters ‘Job’, ‘Job Book_of_Job‘, and ‘Job unfortunate_person unfortunate’ do not seem to create a good understanding of senses for ‘job.n’.

It can also be seen that ‘letter.n’ in Figure 22b does not separate paraphrases in a clean manner. The reason to believe so is that there are two clusters that contain the majority of paraphrase, while the other clusters contain very little amount of paraphrases that are not representatives of a sense. Furthermore, the bottom right cluster contains letters from the Latin and Greek alphabets along with other paraphrases, for instance ‘D’, ‘U’, ‘alpha’, ‘omega’, ‘sigma’ etc. While such cluster depicts the alphabetic meaning of a letter, it also contains words that create noise, for instance ‘block_capital’, ‘descender’, ‘digraph’.

Verbs Figure 23 displays two examples of clusterings for verbs from the WordNet+ dataset. Figure 23a contains a clustering for the verb ‘bring.v’, which we believe separates the word into distinct senses. The senses that can be observed for this word are roughly as follows: throw, bring back, retrieve, make, create and alter. Figure 23b contains a clustering for the verb ‘change.v’, which we believe does not separate the target word into distinct senses. The reason to believe this is not a good representation is similar to the reasons for the clustering of ‘letter.n’ in Figure 22b. While the other seven clusters represent distinct senses such as travel, dress, exchange, replace, and shift, the top and bottom left clusters contain way too many paraphrases in a single cluster. The top left cluster contains 691 paraphrases and it seems that the sense of the cluster is lost, since it contains words like ‘sanitize’, ‘demoralize’, ‘alcoholize’ in the same cluster. This cluster most likely depicts the change in human behaviour along with the change in environment and the world that would make sense to separate.

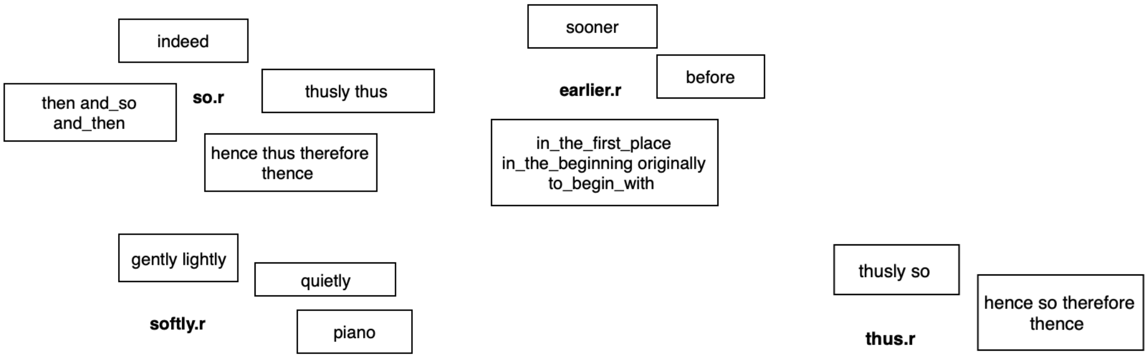

pull pull_in draw_in attract draw
throw_in tinsel lend contribute modify add bestow instill impart factor transfuse alter change
modify alter change
play create work wreak make_for make
land modify alter change
fetch bring_in
get make induce have cause stimulate
bring.v
tube land return whisk channel bring_back transit conduct take_back transmit take convey impart carry ferry fetch transport
institute make create
retrieve get transfer deliver channelise channel convey transmit channelize fetch transport
sensibilize discolor unfit eternalize colour barbarize neutralize strengthen liquidize extemporise tender mechanize waken subdue westernize customise deodourise colonise classicize rarefy
arterialize mix humanize deprave blunt Islamize demist transubstantiate customize elevate evaporate simplify barbarise liquidise energise mythologize oxidate energize clear camp
demagnetize devalue calcify deactivate disaffect clean classicise detransitivize vulgarise round_down officialize thicken industrialise remodel incapacitate mince bring alkalinise exacerbate
personalise transaminate deform impact aggravate conventionalise territorialize communise industrialize brighten destabilize alcoholise digitise stain make_grow plump pressurise exteriorize
lighten smut legitimate tone_down deaden slow_down intensify wake_up amalgamate rusticate humanise chasten heat_up assimilate effeminise cry depolarise flesh_out obscure neutralise set
misdirect allegorize shake vitalize freeze arouse glamourize inseminate corrupt internationalize laicize visualize westernise harshen reform begrime etherealize ornament beef_up blot_out
internationalise beautify vascularize refreshen decorate sexualize oxidize radicalize extend opsonize visualise transpose personalize emulsify cut allegorise animalize animalise reverse
straighten_out disharmonize alcoholize equal contract demulsify concentrate officialise unclutter vesiculate widen color decarboxylate steepen inactivate transfigure cohere supple civilize
coarsen uglify raise regenerate denaturalise invert round_off ease_off mark improve empty enable minify exasperate glamourise dismiss excite change_over harmonize lift wake overturn chill
solvate unify profane sanctify amend sauce lessen ripen plasticise sentimentalize weaken make_clean immaterialize prostrate dynamise magnetize Americanize liberalise soil womanize
bear_on dizzy sanitise ready demoralise saponify iodinate capture demythologise opalise patent make renormalize shorten externalize check uniformize contaminate supercharge deflate
spiritualize mythologise politicise de-emphasize expand transitivise de-emphasise retard mechanise fill_up humble renormalise diversify recombine romanticize restore orientalise chord mingle
fatten_up clot domesticize shape harmonise suburbanise mythicize achromatize touch speed schematize denature mythicise acetylize brutalise isomerise pall add feminise crack habituate
domesticise reinstate match slenderise acetylise normalise awaken militarize depersonalise isomerize change_intensity disorder deconcentrate antique centralize right colour_in redress
desensitize accelerate bolshevise vitrify glamorize solemnise glamorise bolshevize merge untune flatten cloud decentralise moderate turn_back accustom suburbanize achromatise etiolate
lubricate individualise opalize digitalize pressurize eroticize digitalise bear_upon colourise subvert commix equalize indispose individualize muddy decimalize transmute rectify set_up Frenchify
paganise gear_up opacify colorize rationalise bedim prepare condense tame sensibilise fertilise paganize translate bubble perplex form fatten_out virilize desensitise heat rouse dull spice
darken commercialise vaporise dirty centralise temper adorn communize naturalize demonise saturate detransitivise destabilise contribute democratise masculinise nationalise cause_to_sleep
denationalize vivify antiquate animate fill liberalize intransitivise weaponize inspissate demonize spice_up relax overcloud nick demagnetise polarize vulgarize colly draw devilise unscramble
lend depolarize sensify flocculate cool_down replace objectify democratize fasten veil dissonate occidentalize land fertilize defog liquefy unsex obliterate intransitivize change_taste tense
ransom interchange sell barter cash_in exchange transfer stand_in cash trade switch redeem swap fill_in trade_in swop substitute sub
change.v
go transfer travel move locomote
get_dressed dress
incandesce fall_for get_dressed acetylate founder industrialise undulate fall obtain change_shape soften reorient catch settle
alkalinise suburbanise conk_out stabilise go prim find move_up deform move acetylize oxidise keratinise decay ionise promote
brutalise desalinate destabilize change_integrity decalcify feminise crack hydrolyze synthesize arrange spot frost_over shear
mat_up stiffen catch_on crash format acetylise lighten rise swing break fall_in go_bad foul capacitate deaden cool occult dim
indurate sublimate stabilize assimilate die give_out drop feminize refresh roll leave commute isomerize change_surface stratify
break_down vulcanise transmute conform stagnate ionize ulcerate recede relax ascend deoxidise matt-up become_flat fold oxidize
pall pass fail run dawn modernize desalinise introject professionalize glass_over solarise transpire jump separate glaze melt repress
adjust cave_in adapt alkalinize gelatinise sequester vascularize refreshen reticulate climb_up felt mildew hydrolyse resume modify
suburbanize transpose metamorphose flocculate liberalise hydrate caseate come felt_up come_around dry convert mellow_out
animalise reverse receive shift deoxidize modernise lighten_up experience turn solarize change_magnitude regress digest
change_state vesiculate concretize depart divide change_form ice_up color decarboxylate distil matte steepen cool_off opacify
change_posture vitrify isomerise desalinize reflate shallow dilapidate regenerate concretise carbonate transform gradate fold_up
convert exchange capitalise utilize launder replace break commute rectify capitalize
deepen
replace
surf jump cut shift transition leap break switch diphthongize channel-surf diphthongise

(a) 'Good' examples

(b) 'Bad' examples

Figure 23: Examples for verbs from the WordNet+ Gold File

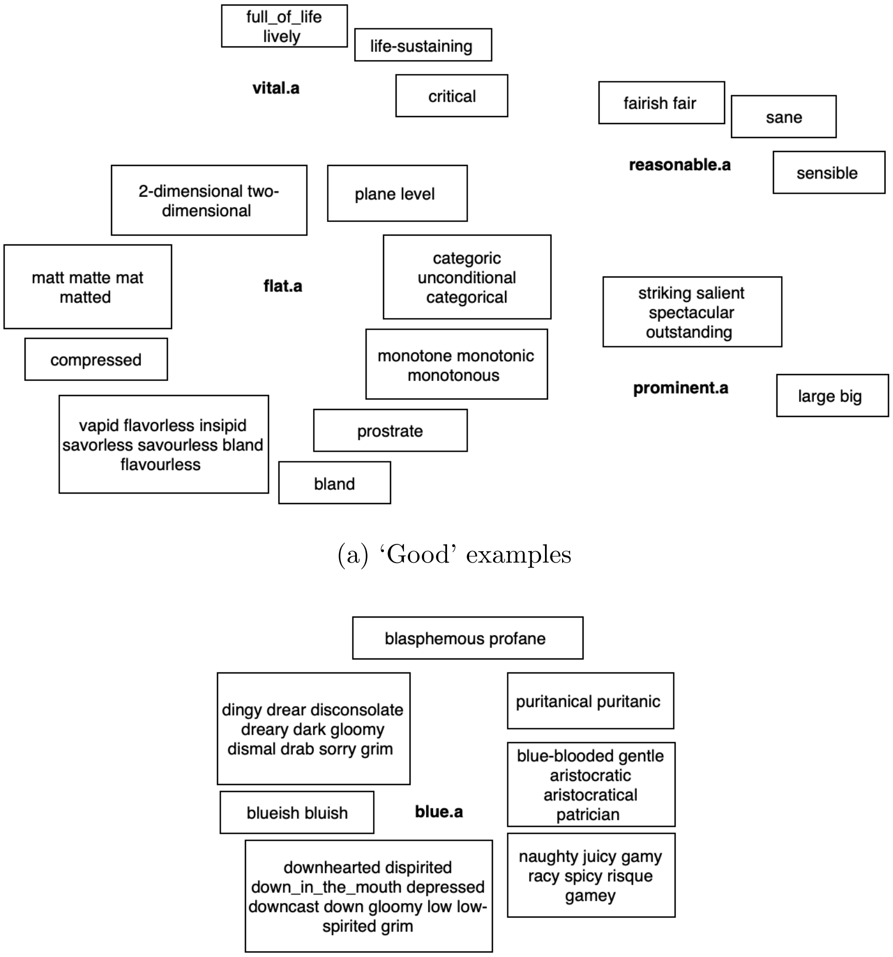
Adverbs Figure 24 displays few examples of clusterings for adverbs from this dataset. Figure 24a contains clusterings for the target word 'so.r', 'softly.r', and 'earlier.r' that seems to separate senses in a clear manner. For instance, for the target word 'earlier.r' there is a differentiation between the concept of time. For the word 'earlier.r' there the absolute and relative difference of time clearly seen in the clustering. Figure 24b contains a clustering for the word ‘thus.r’ that does not seem to differentiate senses. It perhaps would have been more intuitive to combine two clusters into one, but introduce a new cluster where ‘like this’, ‘in this way’, ‘like so’ would have been placed to illustrate ‘thus.r’ in the manner to exemplify something.


indeed
then and_so and_then
so.r
thusly thus
hence thus therefore thence
sooner
before
earlier.r
in_the_first_place in_the_beginning originally to_begin_with
gently lightly
quietly
softly.r
piano
thusly so
hence so therefore thence
thus.r

(a) ‘Good’ examples (b) ‘Bad’ examples

Figure 24: Examples for adverbs from the WordNet+ Gold File

Adjectives Figure 25 displays a few examples of clusterings for the adjectives in this dataset. Based on our observations there are many more adjectives that have been clustered by senses in a clear way in the gold file as seen in Figure 25a. While ‘vital.a’, ‘reasonable.a’, and ‘prominent.a’ contain relatively little amount of paraphrases and perhaps makes the task of separating by senses a little easier, the clustering for ‘flat.a’ has many more words, but the quality of clustering remains high. For instance, ‘flat.a’ has senses that are concerned with the geometrical shape, with the material, with the taste and with the mathematical interpretation of the word. It was relatively hard to find a ‘bad’ example of clustering for the adjectives, however Figure 25b contains one such example for the target word ‘blue.a’. The reason why it might not be a good representative of senses is because there are many outdated senses for the target word, such as ‘puritanical’ or ‘profane’. Furthermore, it seems that the two clusters representing the mood of a person could be combined.


full_of_life lively
life-sustaining
vital.a
critical
fairish fair
sane
reasonable.a
sensible
2-dimensional two-dimensional
plane level
matt matte mat matted
flat.a
categoric unconditional categorical
striking salient spectacular outstanding
compressed
monotone monotonic monotonous
prominent.a
large big
vapid flavorless insipid savorless savourless bland flavourless
prostrate
bland
blasphemous profane
dingy drear disconsolate dreary dark gloomy dismal drab sorry grim
puritanical puritanic
blue-blooded gentle aristocratic aristocratical patrician
blueish bluish
blue.a
downhearted dispirited down_in_the_mouth depressed downcast down gloomy low low-spirited grim
naughty juicy gamy racy spicy risque gamey

(a) ‘Good’ examples

(b) ‘Bad’ examples

Figure 25: Examples for adjectives from the WordNet+ Gold File

4.4.3. Baseline Results

Let us denote the top-scoring Spectral method of Cocos and Callison-Burch (2016) as follows: *Spectral (sm=PPDB2 sc=word2vec e=True)*. This method has entailment enable *(e=True)*, takes PPDB2 scores as a similarity matrix *(sm=PPDB2)*, and silhouette coefficients of word2vec for choosing k *(sc=word2vec)*.

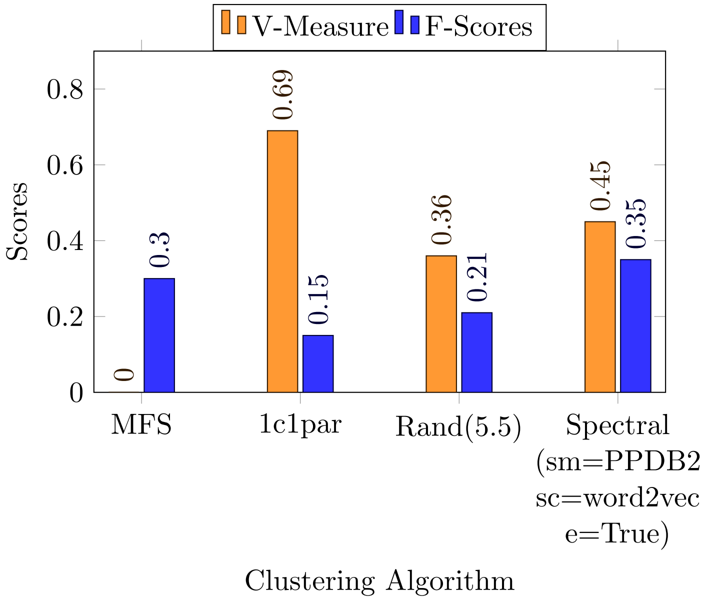
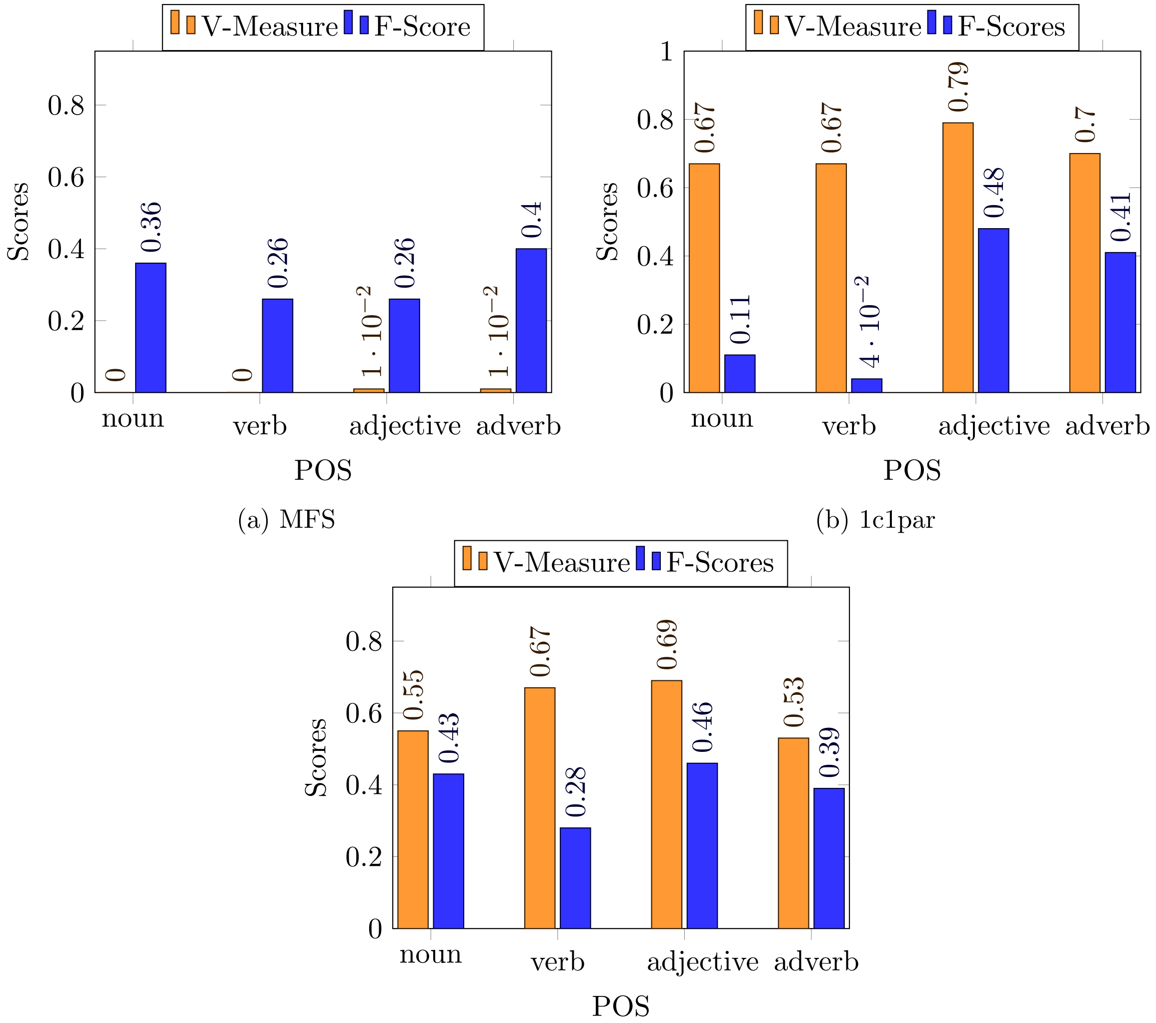


Figure 26: Clustering method performance against WordNet+

Figure 26 displays clustering method performance against the WordNet+ dataset. It can be seen that the results are within 1 standard deviations of the results produced by (Cocos and Callison-Burch, 2016) for all of the baselines and their top performing Spectral method. It can be seen that the performance of the top performing Spectral method by Cocos and Callison-Burch (2016) has the highest F-Score, surpassing MFS. On the other hand, V-Measure is 0.24 less than V-Measure for 1c1par baseline method. Based on the results it seems that the Spectral method has a good balance between the F-Score and V-Measure.



(a) MFS

(b) 1c1par

(c) Spectral (sm=PPDB2 sc=word2vec e=True)

Figure 27: Clustering method performance against WordNet+ separated by POS

Figure 27 displays the of the same algorithms as in Figure 26 broken down by POS. MFS seen in Figure 27a has a a relatively stable performance for F-Score and V-Measure across different types of words. The performance for 1c1par observed in Figure 27b is stable for V-Measure across different types of words, but the F-Score varies significantly, being very low for nouns and verbs, and exceptionally high for adjectives and adverbs. This can indicate that putting a word in its own cluster does not work for nouns and verbs, but works much better for adjectives and adverbs. Finally, based on the performance of Spectral seen in

Figure 27c, the performance of the algorithm relative to baselines is quite stable across POS. The difference between F-Score and V-Measure is relatively small for noun, adjective and adverbs, but is significant for verbs. The results suggest that the strategy for verbs needs to be improved improved in order to improve the overall performance on this dataset.

4.4.4. Results for individual features

Table 20 displays the performance for a subset of combinations of individual similarity measures on WordNet+ dataset. Based on the results from the Table it can be observed that when the entailment is enabled the performance is slightly better, especially for combinations other than PPDB2 as a similarity matrix and word2vec as silhouette coefficients. It seems that with entailment enabled the F-Score and V-Measure go up by 0.1 if image features are used as an input to the similarity matrix. The best performance is still achieved when PPDB2 is used as an input to the similarity matrix and word2vec is used as an input to silhouette coefficients. The other combination that is within 1 standard deviation from the best performing model is when word2vec is used as both an input to the similarity matrix and input to silhouette coefficients.

Similarity Matrix	Silhouette Coefficients	Entail	F-Score	V-Measure	Mean No of Clusters
PPDB2	PPDB2	T	0.361	0.420	3.58
PPDB2	images	T	0.345	0.325	3.10
PPDB2	word2vec	T	0.354	0.449	4.26
contextual	word2vec	T	0.330	0.412	4.21
images	PPDB2	T	0.342	0.379	2.95
images	images	T	0.327	0.275	2.44
images	word2vec	T	0.341	0.411	3.58
word2vec	PPDB2	T	0.344	0.365	2.83
word2vec	images	T	0.330	0.268	2.55
word2vec	word2vec	T	0.340	0.456	4.37
PPDB2	PPDB2	F	0.358	0.419	3.60
PPDB2	images	F	0.343	0.322	3.04
PPDB2	word2vec	F	0.357	0.446	4.10
images	PPDB2	F	0.287	0.222	2.22
images	images	F	0.283	0.207	2.02
images	word2vec	F	0.272	0.289	2.95
word2vec	PPDB2	F	0.330	0.316	2.39
word2vec	images	F	0.318	0.259	1.97
word2vec	word2vec	F	0.288	0.522	5.75

Table 20: WordNet+ performance of Spectral algorithm on a subset of individual features for enabled and disabled entailment

4.4.5. Results for combined features

Table 21 displays the WordNet+ performance of the subset of combinations of individual and combined similarity measures. Based on the results the combination that achieves the best performance is when individual features are used, so when PPDB2 are used as an input to the similarity matrix, and word2vec is used as silhouette coefficients. It seems that combining features performs slightly worse than using individual features alone.

Similarity Matrix	**Silhouette Coefficients**	**F-Score**	**V-Measure**	**Mean No of Clusters**
PPDB2	PPDB2	0.361	0.420	3.58
PPDB2	PPDB2 images	0.358	0.369	3.21
PPDB2	images	0.345	0.325	3.10
PPDB2	word2vec	0.354	0.449	4.26
PPDB2	word2vec PPDB2	0.359	0.437	3.78
PPDB2	word2vec PPDB2 images	0.362	0.388	3.29
PPDB2	word2vec images	0.356	0.350	3.10
PPDB2 contextual	word2vec	0.340	0.422	3.70
PPDB2 images	PPDB2	0.346	0.378	2.87
PPDB2 images	PPDB2 images	0.340	0.318	2.47
PPDB2 images	images	0.332	0.283	2.47
PPDB2 images	word2vec	0.347	0.416	3.53
PPDB2 images	word2vec PPDB2	0.352	0.408	3.13
PPDB2 images	word2vec PPDB2 images	0.347	0.347	2.58
PPDB2 images	word2vec images	0.339	0.304	2.51
PPDB2 images contextual	word2vec	0.341	0.417	3.64
contextual	word2vec	0.330	0.412	4.21
images	PPDB2	0.342	0.379	2.95
images	PPDB2 images	0.336	0.323	2.56
images	images	0.327	0.275	2.44
images	word2vec	0.341	0.411	3.58
images	word2vec PPDB2	0.345	0.406	3.20
images	word2vec PPDB2 images	0.342	0.346	2.63
images	word2vec images	0.338	0.303	2.51
images contextual	word2vec	0.342	0.422	3.67
word2vec	PPDB2	0.344	0.365	2.83
word2vec	PPDB2 images	0.338	0.287	2.42
word2vec	images	0.330	0.268	2.55
word2vec	word2vec	0.340	0.456	4.37
word2vec	word2vec PPDB2	0.352	0.401	3.19
word2vec	word2vec PPDB2 images	0.344	0.330	2.56
word2vec	word2vec images	0.335	0.287	2.57
word2vec PPDB2	PPDB2	0.346	0.383	2.90

word2vec PPDB2	PPDB2 images	0.341	0.324	2.47
word2vec PPDB2	images	0.330	0.286	2.47
word2vec PPDB2	word2vec	0.340	0.428	3.76
word2vec PPDB2	word2vec PPDB2	0.358	0.421	3.20
word2vec PPDB2	word2vec PPDB2 images	0.347	0.352	2.61
word2vec PPDB2	word2vec images	0.338	0.306	2.45
word2vec PPDB2 contextual	word2vec	0.341	0.434	3.81
word2vec PPDB2 images	PPDB2	0.348	0.384	2.88
word2vec PPDB2 images	PPDB2 images	0.340	0.322	2.47
word2vec PPDB2 images	images	0.330	0.285	2.47
word2vec PPDB2 images	word2vec	0.345	0.426	3.69
word2vec PPDB2 images	word2vec PPDB2	0.355	0.412	3.16
word2vec PPDB2 images	word2vec PPDB2 images	0.344	0.350	2.59
word2vec PPDB2 images	word2vec images	0.337	0.307	2.52
word2vec PPDB2 images contextual	word2vec	0.347	0.428	3.73
word2vec contextual	word2vec	0.332	0.445	4.22
word2vec images	PPDB2	0.349	0.384	2.91
word2vec images	PPDB2 images	0.339	0.316	2.50
word2vec images	images	0.329	0.282	2.47
word2vec images	word2vec	0.343	0.425	3.74
word2vec images	word2vec PPDB2	0.356	0.417	3.22
word2vec images	word2vec PPDB2 images	0.346	0.348	2.61
word2vec images	word2vec images	0.338	0.307	2.53
word2vec images contextual	word2vec	0.355	0.426	3.65

Table 21: WordNet+ Performance of Spectral algorithm on a subset of individual and combined features with entailment enabled

4.5. CrowdClusters Dataset

4.5.1. Description

CrowdClusters[3] paraphrase file contains 78 randomly selected target words from the SEMEVAL 2007 dataset, where each target word has a list of paraphrases originating from the unfiltered PPDB2.0 XXL entries. CrowdClusters gold file[4] for this paraphrase file is produced with the help of crowd workers on Amazon Mechanical Turk.

Similarly to Section 4.4, Tables 22, 23 and 24 provide paraphrase and gold files statistics for the CrowdClusters dataset. Table 24 provides information from the gold file about the number of clusters per target word and the number of paraphrases within the corresponding

[3] https://github.com/acocos/cluster_paraphrases/blob/master/data/pp/semeval_tgtlist_rand80_multiword_xxl_PPDB2.0Score_plusself.ppsets

[4] https://github.com/acocos/cluster_paraphrases/blob/master/data/gold/crowd_eval_targets.crowdgold

cluster.

POS	No of target words	No of paraphrases	No of unique paraphrases	Mean No of paraphrases per target word	Median No of paraphrases per target word	Std No of paraphrases per target word
noun	21	4185	2940	199.3	157	137.3
verb	21	8501	4305	404.8	310	269.4
adjective	22	5602	3025	254.6	261	129.9
adverb	14	2541	1151	181.5	163.5	80
all	78	20829	10099	267	214.5	195.5

Table 22: CrowdClusters Paraphrase File POS Breakdown – Number of Paraphrases Statistics

POS	No of target words	No of paraphrases	No of unique paraphrases	Mean No of paraphrases per target word	Median No of paraphrases per target word	Std No of paraphrases per target word
noun	21	148	136	7.1	6	4.7
verb	21	1018	915	48.5	50	20.2
adjective	22	656	598	29.8	23	22.3
adverb	14	286	243	20.5	19	8.6
all	78	2109	1882	27	19	22.6

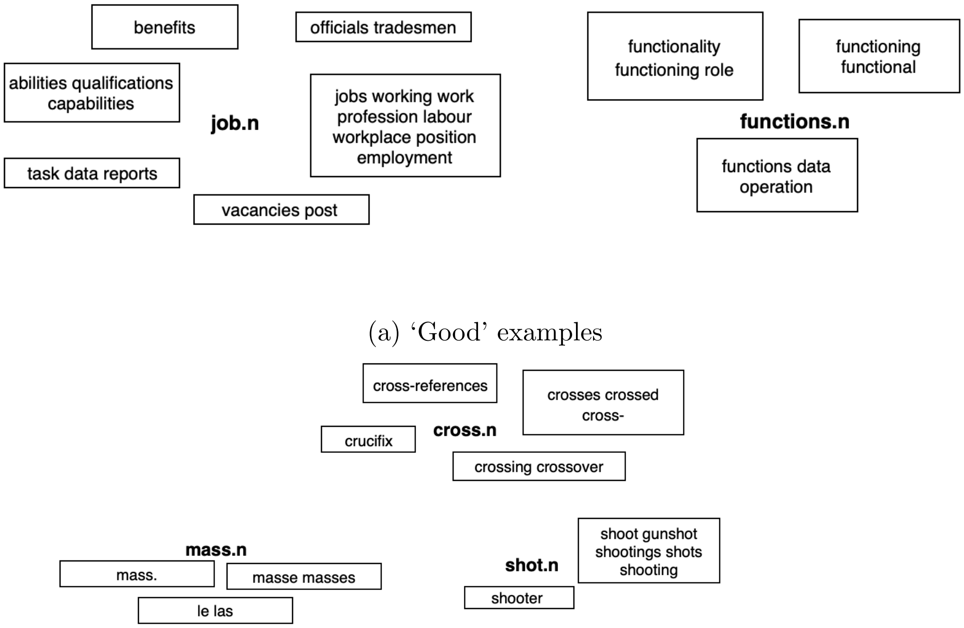
Table 23: CrowdClusters Gold File POS Breakdown – Number of Paraphrases Statistics

POS	Mean No of clusters per target word	Median No of clusters per target word	Std No of clusters per target word	Mean No of paraphrases per cluster	Median No of paraphrases per cluster	Std No of paraphrases per cluster
noun	2.7	2	1.2	2.6	2	1.7
verb	5.5	5	1.7	8.9	5	10.7
adjective	4.4	4	2.4	6.8	4	8.6
adverb	3.7	3.5	1.4	5.5	4	4.7
all	4.1	3.5	2.1	6.6	3	8.5

Table 24: CrowdClusters Gold File POS Breakdown – Number of Clusters and Paraphrases within a Cluster Statistics

Unlike for the WordNet+ dataset, there are many more paraphrases in the paraphrase file as there are in the gold file, most likely justified by the way the gold file was created and the lower number of paraphrases supplied to the crowd workers. It can be seen that there is roughly 10 times more paraphrases in the paraphrase file than there is in the gold file. Similarly to WordNet+ dataset, it can be seen in Tables 22 and 23 that there is a difference in proportion between the mean, median and standard deviation in number of paraphrases for a target word for nouns and verbs. It can also be seen that there are a lot more paraphrases for verbs than there are for adverbs, verbs or nouns, and in fact, the number of paraphrases for a noun is the least from the four POS. Based on Table 24, the number of clusters per target word is around 4.1 across all words, however, the mean and median number of clusters per each POS varies, in particular, with nouns having the least number of clusters and verbs have the most number of clusters. The mean, median, and standard deviation of number of paraphrases per cluster also varies between nouns and verbs, but not as much for adjective and adverbs. Based on these observations it is evident that the algorithm needs to account for different POS.

4.5.2. Gold Clusters Examples

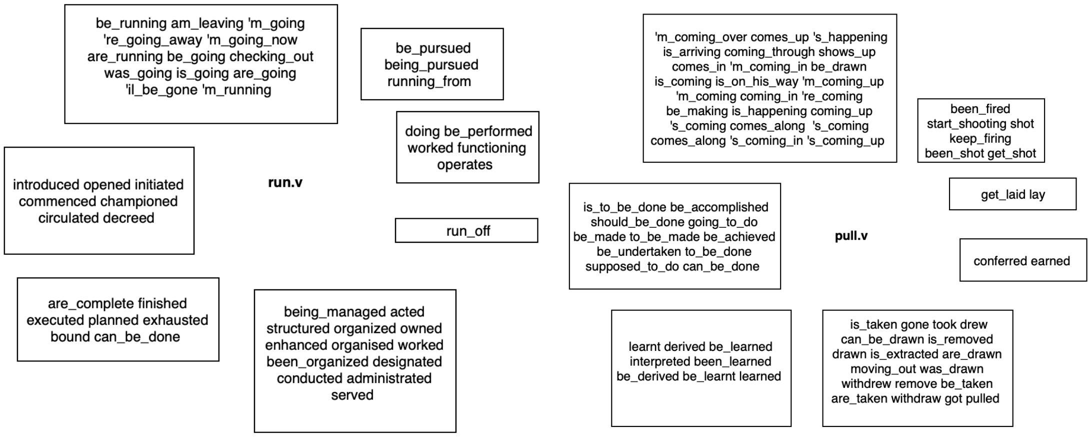


(a) ‘Good’ examples

(b) ‘Bad’ examples

Figure 28: Examples for nouns from the CrowdClusters Gold File

Nouns Figure 28 displays examples of clusterings for some of the nouns from the CrowdClusters gold file. It seems that for nouns ‘job.n’ and ‘functions.n’ seen in Figure 28a some of distinct senses are observed in a clear manner. For instance, for ‘job.n’ the cluster ‘vacancies post’ might refer to the job opening, whereas ‘task data report’ cluster might refer to what a person can produce as part of the job. Figure 28b displays clusterings for nouns ‘mass.n’, ‘shot.n’ and ‘cross.n’, and in our opinion, such clusterings do not represent distinct senses of a polysemous word in a clear manner. Both ‘cross.n’ and ‘mass.n’ have words that have some noise, for instance what can be interpreted as extra punctuation, but more importantly, for ‘mass.n’ more so than for ‘cross.n’ or ‘shot.n’, there is no clear separation of senses. An observation that can be made here is that for some target words, such as ‘mass.n’, there are not enough paraphrases to represent it.

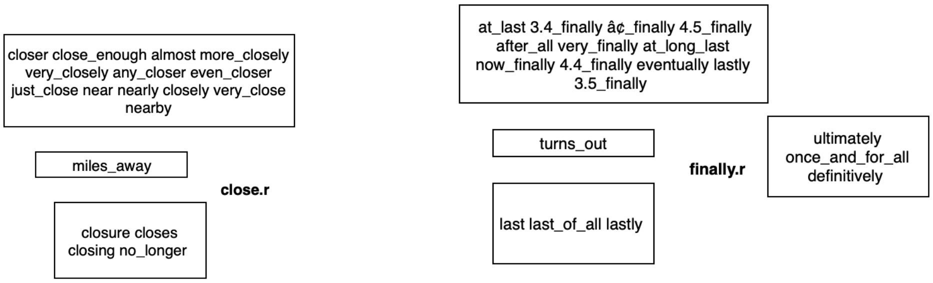

be_running am_leaving 'm_going 're_going_away 'm_going_now are_running be_going checking_out was_going is_going are_going 'il_be_gone 'm_running
be_pursued being_pursued running_from
doing be_performed worked functioning operates
introduced opened initiated commenced championed circulated decreed
run.v
run_off
are_complete finished executed planned exhausted bound can_be_done
being_managed acted structured organized owned enhanced organised worked been_organized designated conducted administrated served
'm_coming_over comes_up 's_happening is_arriving coming_through shows_up comes_in 'm_coming_in be_drawn is_coming is_on_his_way 'm_coming_up 'm_coming coming_in 're_coming be_making is_happening coming_up 's_coming comes_along 's_coming comes_along 's_coming_in 's_coming_up
been_fired start_shooting shot keep_firing been_shot get_shot
get_laid lay
is_to_be_done be_accomplished should_be_done going_to_do be_made to_be_made be_achieved be_undertaken to_be_done supposed_to_do can_be_done
pull.v
conferred earned
learnt derived be_learned interpreted been_learned be_derived be_learnt learned
is_taken gone took drew can_be_drawn is_removed drawn is_extracted are_drawn moving_out was_drawn withdrew remove be_taken are_taken withdraw got pulled

(a) ‘Good’ examples

(b) ‘Bad’ examples

Figure 29: Examples for verbs from the CrowdClusters Gold File

Verbs Figure 29 displays examples of clusterings for some of the verbs from the CrowdClusters gold file. Figure 29a displays clustering for the word ‘run.v’, which seems to have a good quality clusters based on word sense. Some of the senses that can be observed for the word ‘run.v’ are pursuing, managing, physically exercising and finishing. On the other hand, in Figure 29b the target word ‘pull.v’ seems to have low quality verbs more resembling informal context such as ‘re_coming’, ‘s_coming’, ‘’m_coming’ and a lot of paraphrases per cluster.

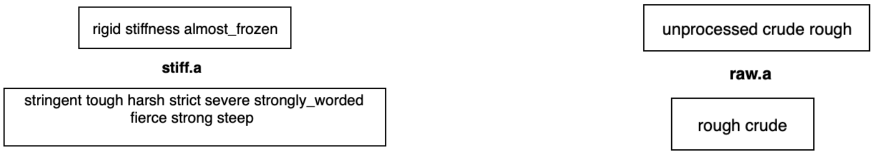

closer close_enough almost more_closely very_closely any_closer even_closer just_close near nearly closely very_close nearby
miles_away
close.r
closure closes closing no_longer
at_last 3.4_finally â¢_finally 4.5_finally after_all very_finally at_long_last now_finally 4.4_finally eventually lastly 3.5_finally
turns_out
finally.r
ultimately once_and_for_all definitively
last last_of_all lastly

(a) ‘Good’ examples

(b) ‘Bad’ examples

Figure 30: Examples for adverbs from the CrowdClusters Gold File

Adverbs Figure 30 displays examples of clusterings for some of the adverbs from the gold file. Figure 30a displays a clustering for the word ‘close.r’, which seems to separate paraphrases in a clear manner. Based on our observation, ‘close.r’ contains the following sense: nearby, no longer and far way. In comparison, for the target word ‘finally.r’ seen in Figure 30b, the cluster at the top contains numbers embedded to words and some foreign characters, which can indicate the defects of the data.


rigid stiffness almost_frozen
stiff.a
stringent tough harsh strict severe strongly_worded fierce strong steep
unprocessed crude rough
raw.a
rough crude

(a) ‘Good’ examples

(b) ‘Bad’ examples

Figure 31: Examples for adjectives from the CrowdClusters Gold File

Adjectives Finally, Figure 31 displays examples of clusterings for some of the adjectives from the gold file. Figure 31a displays a clustering for the word ‘stiff.a’, which seems to have two distinct clusters for a word. The first cluster represents rigid, whereas the second cluster represents sever or strong conditions. In comparison, for the target word ‘raw.a’ in Figure 31a, the cluster at the top contains is basically the cluster at the bottom plus one extra word. Even though ‘raw.a’ can have two distinct senses: ‘rough’ in a sense of conditions and ‘unprocessed’ in a sense of food, those gold file clusters do not make the senses obvious.

4.5.3. Baseline Results

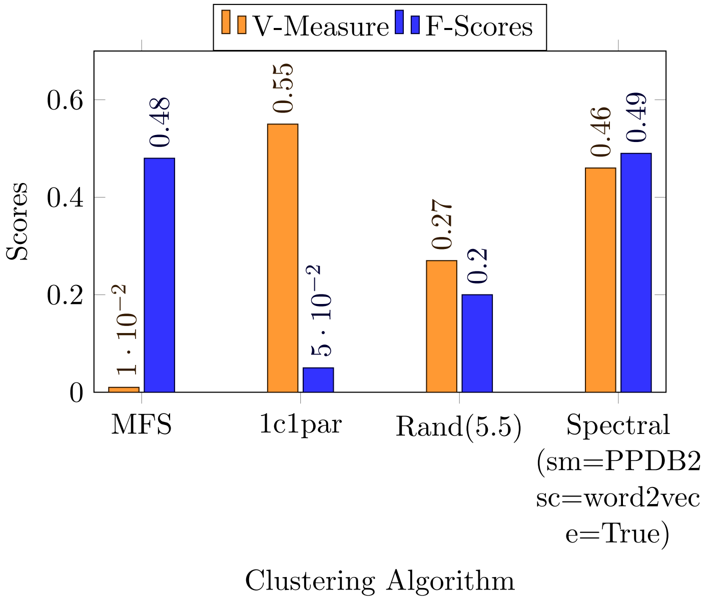
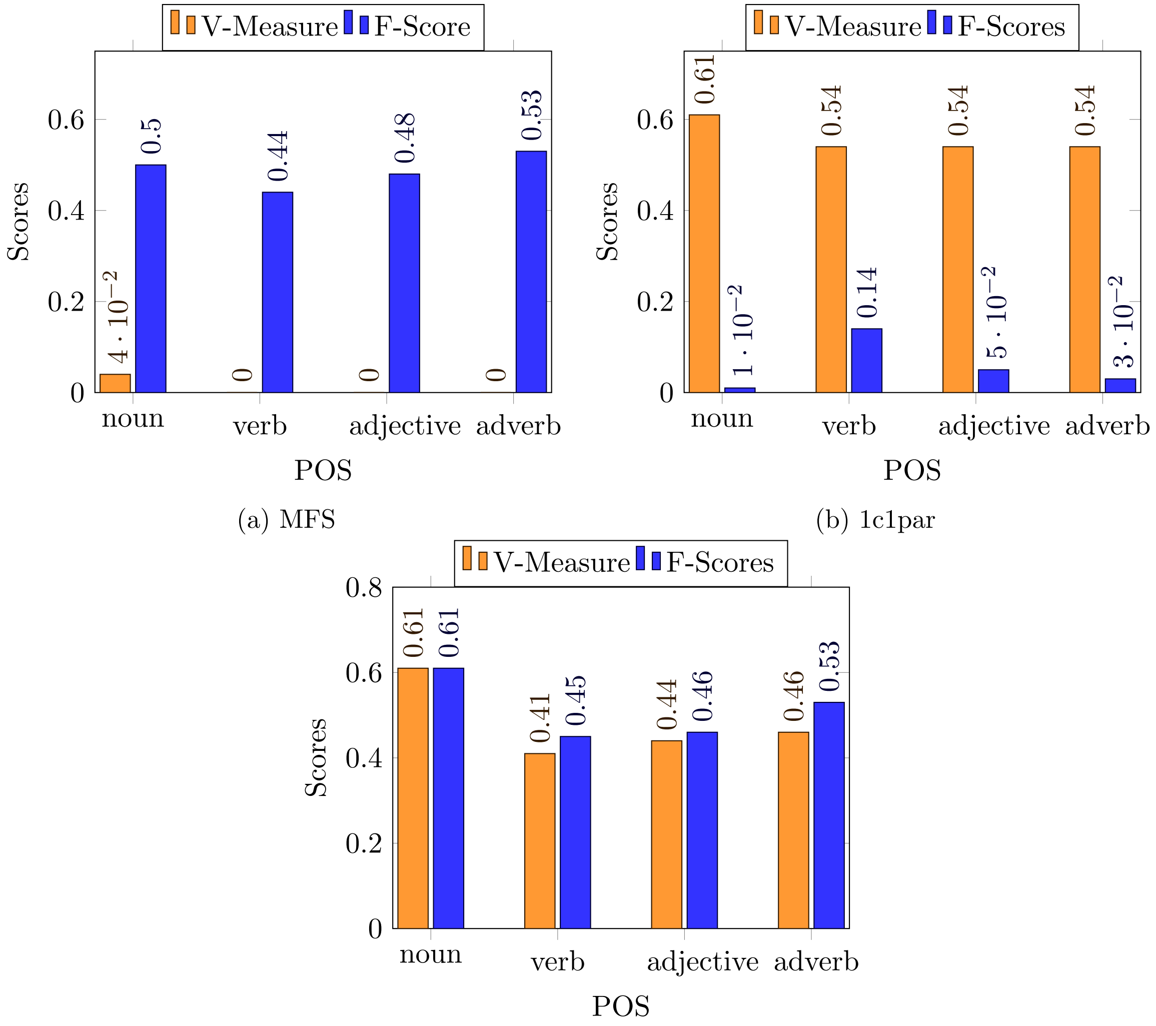


Figure 32: Clustering method performance against CrowdClusters

Figure 32 denotes clustering method performance against tis dataset. It can be seen that the results are within 2 standard deviations of the results seen in the paper (Cocos and Callison-Burch, 2016) for all of the baselines and their top performing Spectral method. Based on the results, MFS method has a relatively high F-Score and a low V-Measure, whereas 1c1par method has a relatively high V-Measure and a low F-Score. The performance of the top performing Spectral method by Cocos and Callison-Burch (2016) has a relatively high V-Measure similar to 1c1par and a relatively high F-Score similar to MFS.



(a) MFS

(b) 1c1par

(c) Spectral (sm=PPDB2 sc=word2vec e=True)

Figure 33: Clustering method performance against CrowdClusters separated by POS

Measure	**Score**
Mean No of Gold Clusters	4.10
Mean No of Solution Clusters	4.63
Std No of Solution Clusters	2.81
Mean of |No of Gold Clusters - No of Solution Clusters|	1.77
Std of |No of Gold Clusters - No of Solution Clusters|	2.22

Table 25: Statistics on the number of gold clusters and predicted clusters for Spectral (sm=PPDB2 sc=word2vec e=True) on CrowdClusters

Figure 33 displays clustering method performance of the same baselines as in Figure 32, but separated by POS. Based on the performance of MFS in Figure 33a, V-Measure is 0 for verbs, adjective and adverbs and slightly higher for nouns, while F-Score varies at most by 0.9 between the POS, being the lowest for verbs and highest for adverbs. Based on the performance of 1c1par in Figure 33b, there are slight differences between V-Measure and F-Score across all present POS. Nouns for this method has the highest V-Measure and the lowest F-Score, while verbs have the highest F-Score and second highest V-Measure. Finally, based on the performance of Spectral seen in Figure 33c and Table, the performance of the algorithm varies within POS greatly. Verbs have the lowest paired F-Score and V-Measure, followed by adjectives and adverbs, while nouns have the highest paired F-Score and V-Measure. Based on Figure 33c, there is a need to treat POS differently when performing clustering. Table 25 displays the basic statistics of the number of gold clusters and predicted clusters. It can be seen from the Table that the mean difference between the number of gold clusters and predicted clusters is 1.77, which indicates that the algorithm predicts the number of in accordance to the gold clusters.

4.5.4. Results for individual features

Table 27 displays the CrowdClusters performance for a subset of combinations of individual similarity measures. It can be seen that when the entailment is enabled the performance becomes better by a slight margin. Similarly to WordNet+ dataset, based on this table the best performance is still achieved when PPDB2 is used as an input to the similarity matrix and word2vec is used as an input to silhouette coefficients. Other combinations that do well on this dataset are contextual or word2vec as similarity matrix and word2vec as an input to silhouette coefficients.

Similarity Matrix	**Silhouette Coefficients**	**Entail**	**F-Score**	**V-Measure**	**Mean No of Clusters**
PPDB2	PPDB2	T	0.497	0.427	4.05
PPDB2	images	T	0.492	0.388	4.10
PPDB2	word2vec	T	0.493	0.463	4.63
contextual	word2vec	T	0.455	0.405	4.65
images	PPDB2	T	0.123	0.551	23.24

images	images	T	0.126	0.550	23.05
images	word2vec	T	0.122	0.551	23.17
word2vec	PPDB2	T	0.513	0.380	2.78
word2vec	images	T	0.497	0.328	2.81
word2vec	word2vec	T	0.470	0.432	4.26
PPDB2	PPDB2	F	0.497	0.429	4.08
PPDB2	images	F	0.497	0.385	4.04
PPDB2	word2vec	F	0.495	0.435	4.18
contextual	word2vec	F	0.322	0.476	9.52
images	PPDB2	F	0.118	0.546	23.24
images	images	F	0.121	0.545	23.06
images	word2vec	F	0.120	0.547	23.12
word2vec	PPDB2	F	0.511	0.356	2.30
word2vec	images	F	0.495	0.303	2.04
word2vec	word2vec	F	0.420	0.483	5.57

Table 26: CrowdClusters performance of Spectral algorithm on a subset of individual features for enabled and disabled entailment

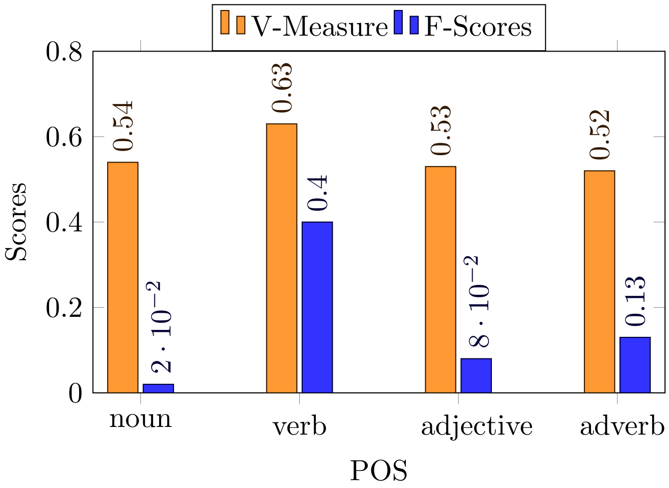


Figure 34: POS performance breakdown for Spectral (sm=images sc=word2vec e=True) configuration from Table 26

It can also be seen that the performance of images as an input to the similarity matrix is very low in the F-Score and high in V-Measure. The mean number of clusters for such configuration is very high, which can indicate that there is a problem with inferring the number of clusters for certain words. This observation is further confirmed in Figure 34 when there is a breakdown by POS. It can be seen that the algorithm produces good results for verbs, but behaves like 1c1par for adjectives, adverbs and nouns. Such results indicate the need to tune the algorithm for different types of datasets and different similarity measures.

4.5.5. Results for combined features

Table 27 displays the CrowdClusters performance of the subset of combinations of individual and combined similarity measures described in Section 4.3.4. Based on the results the combination that achieves the best performance is when PPDB2 and images are used as an input to the similarity matrix, and word2vec is used as silhouette coefficients. This performance is slightly better than the performance of just PPDB as an input to the similarity matrix. It seems that for CrowdClusters dataset the combining features yields a higher performance as supposed to combining features for the WordNet+ dataset.

Similarity Matrix	**Silhouette Coefficients**	**F-Score**	**V-Measure**	**Mean No of Clusters**
PPDB2	PPDB2	0.497	0.427	4.05
PPDB2	PPDB2 images	0.506	0.399	3.92
PPDB2	images	0.492	0.388	4.10
PPDB2	word2vec	0.493	0.463	4.63
PPDB2	word2vec PPDB2	0.499	0.441	4.15
PPDB2	word2vec PPDB2 images	0.508	0.408	3.90
PPDB2	word2vec images	0.495	0.400	4.19
PPDB2 contextual	word2vec	0.502	0.410	3.55
PPDB2 images	PPDB2	0.501	0.426	4.01
PPDB2 images	PPDB2 images	0.508	0.401	3.92
PPDB2 images	images	0.499	0.391	4.04
PPDB2 images	word2vec	0.503	0.466	4.54
PPDB2 images	word2vec PPDB2	0.504	0.445	4.14
PPDB2 images	word2vec PPDB2 images	0.511	0.410	3.90
PPDB2 images	word2vec images	0.503	0.404	4.14
PPDB2 images contextual	word2vec	0.504	0.409	3.50
contextual	word2vec	0.455	0.405	4.65
images	PPDB2	0.123	0.551	23.24
images	PPDB2 images	0.123	0.551	23.24
images	images	0.126	0.550	23.05
images	word2vec	0.122	0.551	23.17
images	word2vec PPDB2	0.123	0.551	23.24
images	word2vec PPDB2 images	0.123	0.551	23.21
images	word2vec images	0.126	0.550	23.04
images contextual	word2vec	0.416	0.414	6.52
word2vec	PPDB2	0.513	0.380	2.78
word2vec	PPDB2 images	0.520	0.331	2.37
word2vec	images	0.497	0.328	2.81
word2vec	word2vec	0.470	0.432	4.26
word2vec	word2vec PPDB2	0.514	0.402	2.95
word2vec	word2vec PPDB2 images	0.521	0.365	2.62
word2vec	word2vec images	0.519	0.351	2.65
word2vec PPDB2	PPDB2	0.503	0.368	2.88

word2vec PPDB2	PPDB2 images	0.512	0.323	2.37
word2vec PPDB2	images	0.498	0.321	2.73
word2vec PPDB2	word2vec	0.513	0.407	3.27
word2vec PPDB2	word2vec PPDB2	0.513	0.407	2.99
word2vec PPDB2	word2vec PPDB2 images	0.518	0.349	2.54
word2vec PPDB2	word2vec images	0.512	0.339	2.59
word2vec PPDB2 contextual	word2vec	0.508	0.410	3.28
word2vec PPDB2 images	PPDB2	0.501	0.367	2.86
word2vec PPDB2 images	PPDB2 images	0.513	0.322	2.35
word2vec PPDB2 images	images	0.498	0.320	2.71
word2vec PPDB2 images	word2vec	0.496	0.401	3.36
word2vec PPDB2 images	word2vec PPDB2	0.510	0.403	2.95
word2vec PPDB2 images	word2vec PPDB2 images	0.518	0.349	2.54
word2vec PPDB2 images	word2vec images	0.512	0.339	2.59
word2vec PPDB2 images contextual	word2vec	0.503	0.409	3.37
word2vec contextual	word2vec	0.462	0.419	4.01
word2vec images	PPDB2	0.505	0.370	2.78
word2vec images	PPDB2 images	0.514	0.320	2.33
word2vec images	images	0.494	0.321	2.78
word2vec images	word2vec	0.465	0.421	4.17
word2vec images	word2vec PPDB2	0.503	0.394	2.99
word2vec images	word2vec PPDB2 images	0.513	0.354	2.62
word2vec images	word2vec images	0.508	0.343	2.68
word2vec images contextual	word2vec	0.464	0.423	4.18

Table 27: CrowdClusters performance of Spectral algorithm on a subset of individual and combined features with entailment enabled

4.6. WordNet+ Gold 2.0 Dataset

4.6.1. Motivation: Limitations of existing datasets and other problems

Previous sections on WordNet+ and CrowdClusters datasets, specifically Sections 4.4.2 and 4.5.2, outlined the examples of clusterings for each part-of-speech from gold files that we believe might not cluster distinct senses for a target word in a clear manner. The most common problem for the 'bad' examples included clustering too many paraphrases in the same cluster, which made the cluster loose its sense. Another problems included outliers, clear word mistakes, almost complete repetition of clusters and so on. Based on the qualitative observations it is evident that there exist target words for which gold clusterings are not of a high quality.

However, if we assume that the quality of a gold file is high, there is one main limitation we found in implementation of Cocos and Callison-Burch (2016). It has to do with the way the evaluation is done on WordNet+ and CrowdClusters datasets. In order to evaluate the performance of an algorithm an intersection of predicted clustering from the input paraphrase file and gold clustering from the output gold file needs to be performed. The evaluation is not able to take into account unknown words. We refer to such operation as *post-filtering*. As a consequence, taking an intersection produces a low quality clustering. In order to illustrate this problem, let us take a real example from the WordNet+ dataset for the target word 'saint.n'. The input paraphrase file referenced in Section 4.4 contains 'saint.n' that has 3 paraphrases: god, angel, saint. Those 3 paraphrases need to be clustered by senses. In the gold file referenced in Section 4.4 'saint.n' has 3 clusters that can be seen in Figure 35.

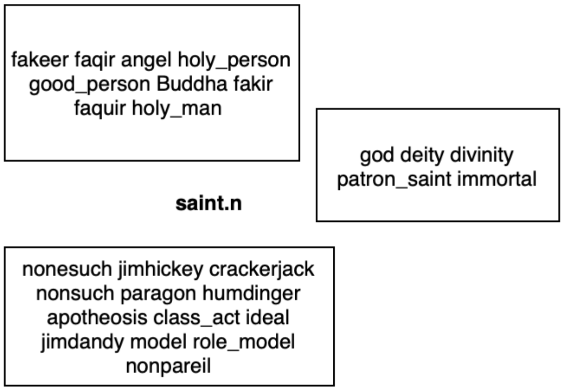

fakeer faqir angel holy_person good_person Buddha fakir faquir holy_man
god deity divinity patron_saint immortal
saint.n
nonesuch jimhickey crackerjack nonsuch paragon humdinger apotheosis class_act ideal jimdandy model role_model nonpareil

Figure 35: WordNet+ gold file clustering of target word 'saint.n'

The Spectral method (with PPDB2 scores as an input to the similarity matrix and word2vec scores as an silhouette coefficients) clusters each of the three paraphrases into 3 distinct clusters, acting as 1c1par baseline. In order to evaluate the clustering, the predicted clustering is compared against gold clustering, where only intersected words get evaluated.

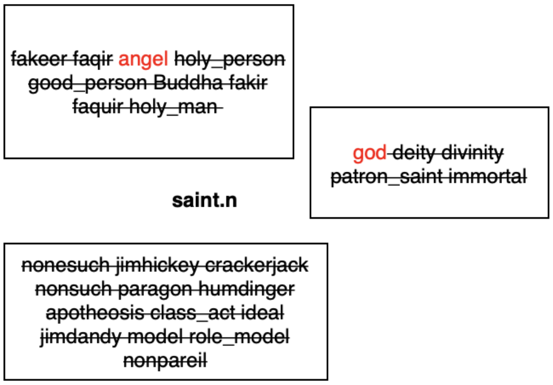

~~fakeer faqir~~ angel ~~holy_person~~
~~good_person Buddha fakir~~
~~faquir holy_man~~
god ~~deity divinity~~
~~patron_saint immortal~~
saint.n
~~nonesuch jimhickey crackerjack~~
~~nonsuch paragon humdinger~~
~~apotheosis class_act ideal~~
~~jimdandy model role_model~~
~~nonpareil~~

Figure 36: WordNet+ gold file clustering of target word ‘saint.n’ post-filtering

That means that the word ‘saint’ is removed from the predicted clustering since it does not occur in the gold clustering, and the words ‘fakeer, faqir, holy_person, good_person, Buddha, fakir, faquir, holy_man, nonesuch, jimhickey, crackerjack, nonsuch, paragon, humdinger, apotheosis, class_act, ideal, jimdandy, model, role_model, nonpareil, deity, divinity, patron_saint, and immortal’ also get removed from the gold clustering since they do not occur in the paraphrase file and cannot be evaluated. As observed in Figure 36, the post-filtering of gold clustering produces just 2 clusters with the words ‘god’ and ‘angel’ being in separate clusters. The post-filtering of predicted clustering produces also 2 clusters with ‘god’ and ‘angel’ being in separate clusters. The F-Score and V-Measure are both predicted to be 1, thus being a perfect match between predicted and gold clustering. What this evaluation for ‘saint.n’ fails to observe is that 33% was removed from the paraphrase file and 93% was removed from the gold file. Could the gold clustering be trusted if 93% of the data is removed?

4.6.2. Generation

In order to generate this new WordNet+ Gold 2.0 dataset, we have disregarded the paraphrase file described in Section 4.4 and only taken the WordNet+ Gold file[5]. We then removed all words in the gold file for which there are no image representation, since word2vec

[5] https://github.com/acocos/cluster_paraphrases/blob/master/data/gold/wordnet_eval_targets.wngold

score can be obtained for unknown words, but the image representation cannot (we use static version of the dataset (Callahan, 2017)).

As a result, from 8667 unique words that occur in the WordNet+ gold file 3006 were removed, which accounts for around 35%. Over 60% of those removed words contained at least one underscore or a dash. Some of the examples from the removed words are: rake_off, charge_per_unit, business_enterprise, Agenise, progress_to, balaclava, deglycerolize, chopping_board, cornice, go_wrong and jump_off. Therefore, for each target word and clustering of a target word in the filtered gold file we generated the input paraphrase file but extracted words from the clustering and aggregated them into a single list. The new gold file, paraphrase file and words that were removed are available to view offline. As a side note, we decided to completely disregard PPDB2 scores as a similarity measure since there was no data for more than 50% of the paraphrases from WordNet+ dataset. Even though PPDB2 scores as an input to the similarity matrix score was the top performing algorithm for both WordNet+ and CrowdClusters datasets, we believe that word2vec as a similarity measure had a comparable performance within 1 standard deviation of PPDB2 score. We justify the high performance of PPDB2 score due to but sparse but strong signal for the clustering of paraphrases.

4.6.3. Description

Tables 28 and 29 provide paraphrase and gold files statistics for the WordNet+ dataset, in particular the number of paraphrases and clusters per target word broken down by POS.

POS	**No of target words**	**No of paraphrases**	**No of unique paraphrases**	**Mean No of paraphrases per target word**	**Median No of paraphrases per target word**	**Std No of paraphrases per target word**
noun	60	2752	2320	45.9	45	28.5
verb	52	5797	3056	111.5	80.5	98.911
adjective	54	616	527	11.4	9	7.3
adverb	35	242	194	6.9	7	3.2
all	201	9407	5661	46.8	20	66.9

Table 28: WordNet+ Gold 2.0 Paraphrase and Gold File POS Breakdown – Number of Paraphrases Statistics

POS	**Mean No of clusters per target word**	**Median No of clusters per target word**	**Std No of clusters per target word**	**Mean No of paraphrases per cluster**	**Median No of paraphrases per cluster**	**Std No of paraphrases per cluster**
noun	7.6	7	4.6	6.4	4	8.2
verb	15.9	11.5	11	8.3	5	18.8
adjective	5.2	4.5	2.9	2.3	2	1.9
adverb	3	3	1.2	2.4	2	1.8
all	8.3	6	7.8	6.4	3	14.1

Table 29: WordNet+ Gold 2.0 Gold File POS Breakdown – Number of Clusters and Paraphrases within a Cluster Statistics

Based on Table 28 the number of paraphrases per target word varies for different POS, where verbs have the largest number and adverbs have the smallest number. According to Table 29 the number of clusters per target word is around 8.3 across all words, is just 0.2 smaller than the number of clusters per target word in the the original gold file seen in Table 19. Similarly to the original dataset, the mean and median number of clusters per each POS for the new dataset varies significantly with verbs having the most number of clusters and adverbs having the least number of clusters. One other observation is that the statistics for the number of paraphrases per cluster for adverbs and adjectives are almost identical even though there are three times as much paraphrases for adjectives as there are for adverbs, which is reflected in the number of clusters per target word. The conclusion that can be derived from these observations is that the algorithm needs to account for different POS tag, since the distribution of paraphrases and clusters per target word varies greatly between different types of words.

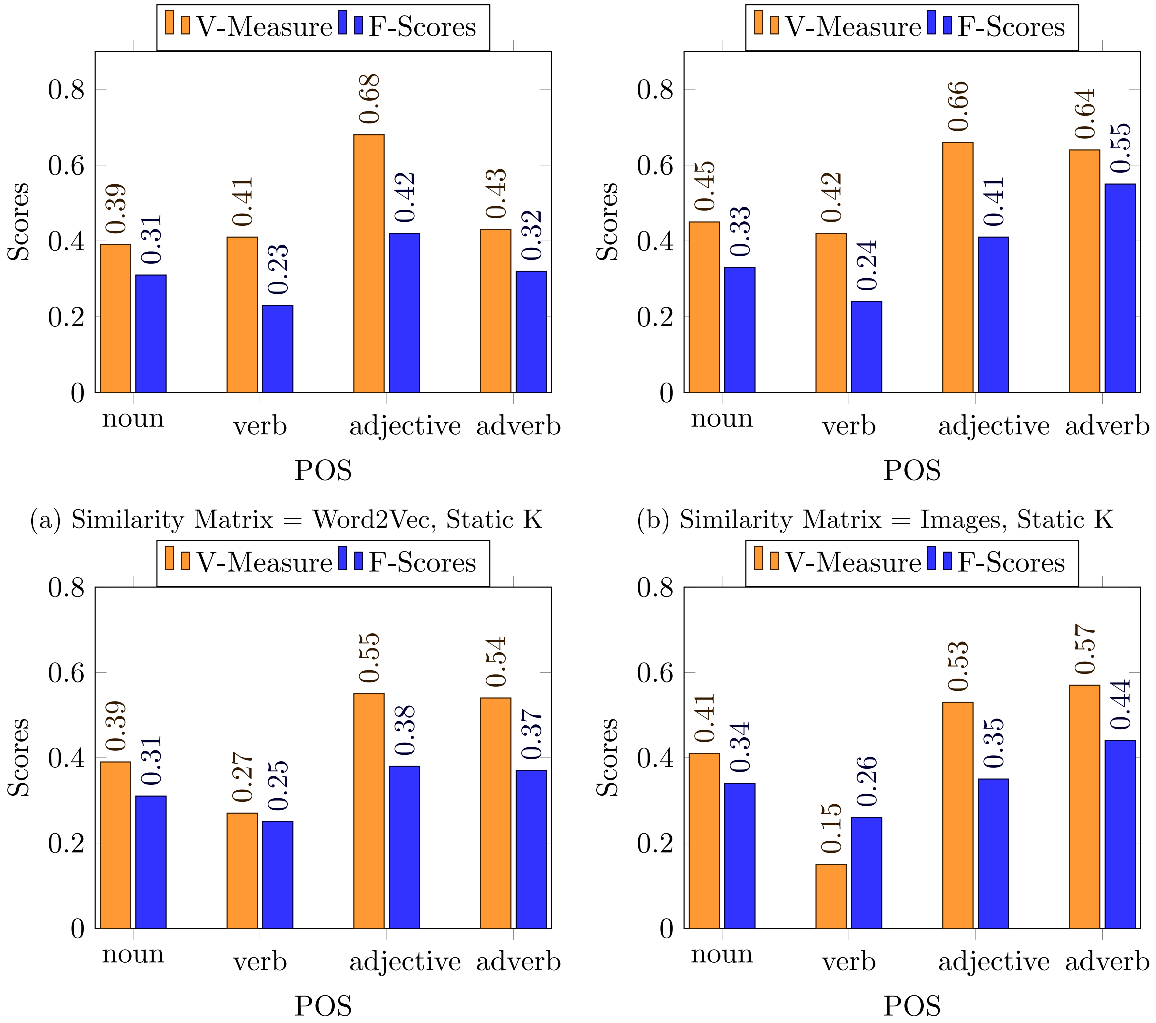
4.6.4. Results

Table 30 displays the clustering performance of baselines along with the top models against the WordNet+ Gold 2.0 dataset.

Clustering Method		**F-Score**	**V-Measure**
Baselines			
MFS		0.31	0.0
1c1Par		0.08	0.65
RAND		0.19	0.27
Similarity Matrix	**K**		
Images	Static	0.32	0.49
Word2vec	Static	0.31	0.48
Word2vec, Images	Static	0.31	0.49
Images	Word2vec	0.31	0.32
Word2vec	Word2vec	0.3	0.37
Images	Images	0.31	0.22
Word2vec	Images	0.31	0.25
Wordvec, Images	Word2vec	0.3	0.32
Wordvec, Images	Images	0.31	0.26

Table 30: Clustering method performance of baseline models and top models of Spectral algorithm against the WordNet+ Gold 2.0 dataset

Based on the results for the baseline, MFS achieves an F-Score of 0.31 and V-Measure of 0, while 1c1Par achieves an F-Score of 0.08 and V-Measure of 0.65. The performance of RAND is much more balanced in comparison to MFS or 1c1Par with an F-Score of 0.27 and V-Measure of 0.19. When k is known ahead of time based on the gold clusterings (static), the performance of an algorithm is higher in V-Measure by around 0.15, as supposed to when k is inferred using silhouette coefficients. The performance of image features and word2vec is equivalent for when k is static, and in fact the performance of image features is slightly better. However, hen k is determined using silhouette coefficients, the performance of image features is slightly worse when word2vec is used as silhouette coefficients and significantly worse when image features are used as silhouette coefficients.



(a) Similarity Matrix = Word2Vec, Static K

(b) Similarity Matrix = Images, Static K

(c) Similarity Matrix = Word2Vec, Silhouette Coefficients = Word2Vec

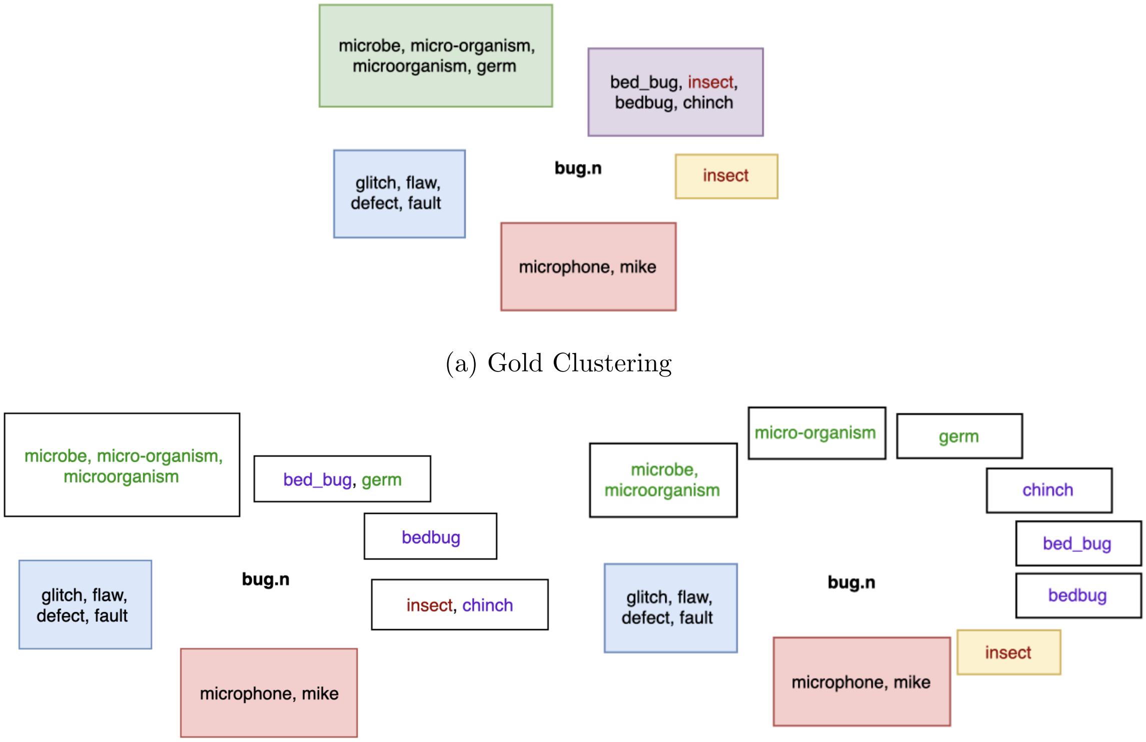
(d) Similarity Matrix = Images, Silhouette Coefficients = Word2Vec

Figure 37: Clustering method performance of four models against WordNet+ Gold 2.0 separated by POS

An interesting insight into how image features differ from word2vec can be observed in Figure 37 for the two approaches of selecting k. When static k is used, the performance of image features is higher for nouns and adverbs and is lower for verbs and adjectives in comparison to word2vec. This can be explained by the inherent nature of the images. Same phenomena is observed when k is chosen based on word2vec as silhouette coefficients. What is important to note is that the combination of image features as similarity matrix and word2vec as silhouette coefficients is worse by 0.05 in V-Measure as seen in Table 30, a direct impact of a drop in V-Measure for verbs.

4.6.5. Gold and Predicted Clusters Examples

The following section contains examples of predicted and gold clusters for nouns, adjectives, verbs, and adverbs. Please note that the word ‘insect’ appears in multiple clusters according to the gold file and thus is highlighted with burgundy colour. Words inside the predicted clusters are highlighted with the colour of the gold cluster other than the words in the burgundy colour.


microbe, micro-organism, microorganism, germ
bed_bug, insect, bedbug, chinch
bug.n
glitch, flaw, defect, fault
insect
microphone, mike
microbe, micro-organism, microorganism
bed_bug, germ
bedbug
bug.n
glitch, flaw, defect, fault
insect, chinch
microphone, mike
micro-organism
germ
microbe, microorganism
chinch
bed_bug
bedbug
bug.n
glitch, flaw, defect, fault
insect
microphone, mike

(a) Gold Clustering

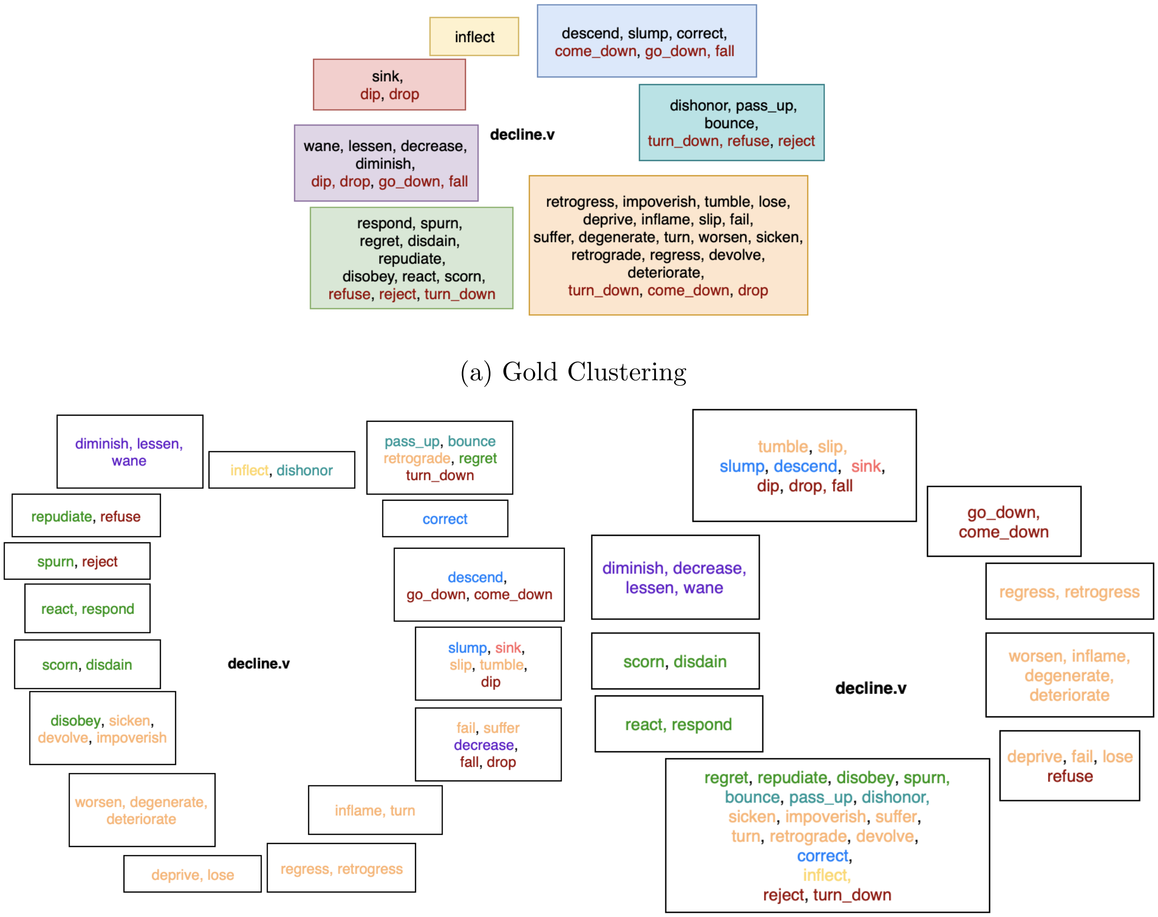
(b) Similarity Matrix = Images, Silhouette Coefficients = Word2vec,
F-Score: 0.667, V-Measure: 0.846

(c) Similarity Matrix = Word2vec, Silhouette Coefficients = Word2vec,
F-Score: 0.529, V-Measure: 0.643

Figure 38: Gold clustering along with the predicted clusterings for the word ‘bug.n’

Nouns Figure 38 displays a gold clustering along with the predicted clusterings for the word 'bug.n'. The clustering algorithm performs hard clustering and thus is unable to place a word into multiple clusters. In this example image features achieve higher performance in both F-Score and V-Measure when compared to word2vec as an input to the similarity matrix. It can be seen that both of the predicted clusterings are quite similar to each other in the way they cluster paraphrases. Both configurations cluster blue and red clusters perfectly. They both misplace the words 'bed_bug' and 'bedbug' in different clusters and cannot cluster them with 'chinch'. Image features place 'micro-organism' and 'microorganism' together, while word2vec separates them. Based on the figure it seems that both configurations have trouble placing 'microorganism', 'bedbug', 'germ', and 'chinch'.

Verbs Figure 39 displays a gold clustering along with the predicted clusterings for the word 'decline.v'. There are lots of words that occur in multiple clusters, such as 'go_down', 'fall', 'drop', 'reject' and so on. More importantly, there a lot more paraphrases within a cluster, which is consistent with observations made in Table 29. Predicted clusterings achieve a lower F-Score and V-Measure relative to the previous example of the word *bug.n* seen in Figure 38. Word2vec as a similarity matrix achieves a more balanced performance on both evaluation metrics. It seems that images features and word2vec have at least 3 predicted clusters that are equivalent. for example 'react' and 'respond' are placed together, similarly to 'regress' and 'retrogress'. Furthermore, image features separate paraphrases in 16 clusters, 7 more clusters than word2vec. Both configurations have clusters that have a lot of paraphrases from different clusters, as can be seen by a variation of colours inside boxes. While there is little confused for the purple cluster, there seems to be a lot more problem in clustering paraphrases from orange, green and blue clusters.



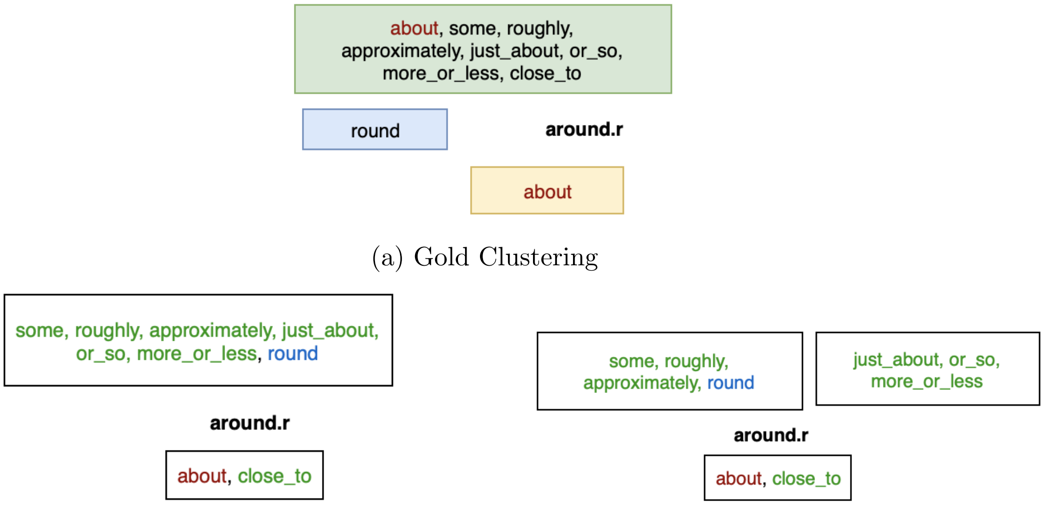
(a) Gold Clustering

(b) Similarity Matrix = Images, Silhouette Coefficients = Word2vec,
F-Score: 0.18, V-Measure: 0.593

(c) Similarity Matrix = Word2vec, Silhouette Coefficients = Word2vec,
F-Score: 0.307, V-Measure: 0.403

Figure 39: Gold clustering along with the predicted clusterings for the word 'decline.v'

Adverbs Figure 40 shows gold and predicted clusterings for the word 'around.a'. The gold clustering contains a paraphrase 'about' that is placed as its own cluster, but also inside a green cluster. A clear distinction of senses can be observed for this target word. Image features achieve a significantly higher performance in F-Score and similar performance n V-Measure. Based on the predictions, image features seem to to a better job in bring paraphrases that belong to the green cluster together. What is interesting to see is that 'close_to' is being placed with 'about' in both configurations. Both configurations can't separate the word 'round' from the paraphrases in green cluster such as 'some', 'approximately', and ‘roughly’.

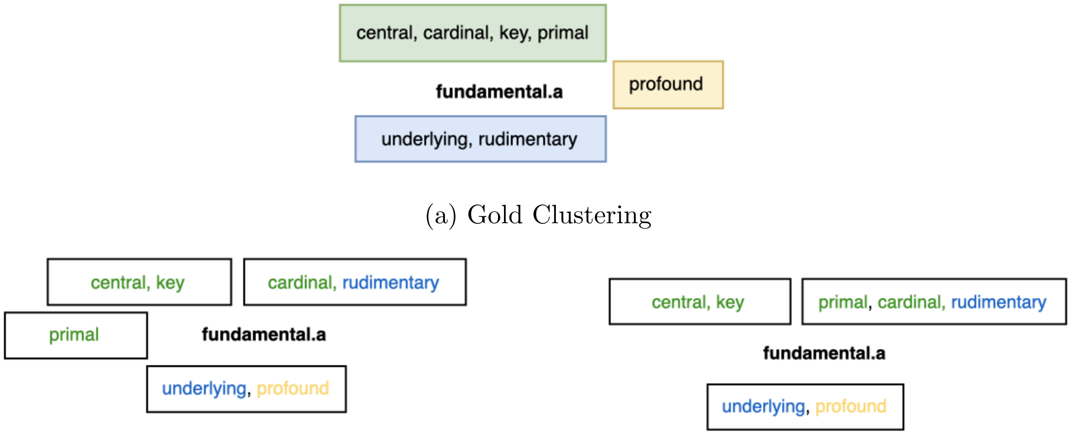

about, some, roughly, approximately, just_about, or_so, more_or_less, close_to
round
around.r
about
some, roughly, approximately, just_about, or_so, more_or_less, round
around.r
about, close_to
some, roughly, approximately, round
just_about, or_so, more_or_less
around.r
about, close_to

(a) Gold Clustering

(b) Similarity Matrix = Images, Silhouette Coefficients = Word2vec,
F-Score: 0.615, V-Measure: 0.727

(c) Similarity Matrix = Word2vec, Silhouette Coefficients = Word2vec,
F-Score: 0.35, V-Measure: 0.7

Figure 40: Gold clustering along with the predicted clusterings for the word ‘around.r’



(a) Gold Clustering

(b) Similarity Matrix = Images, Silhouette Coefficients = Word2vec,
F-Score: 0.167, V-Measure: 0.25

(c) Similarity Matrix = Word2vec, Silhouette Coefficients = Word2vec,
F-Score: 0.308, V-Measure: 0.4

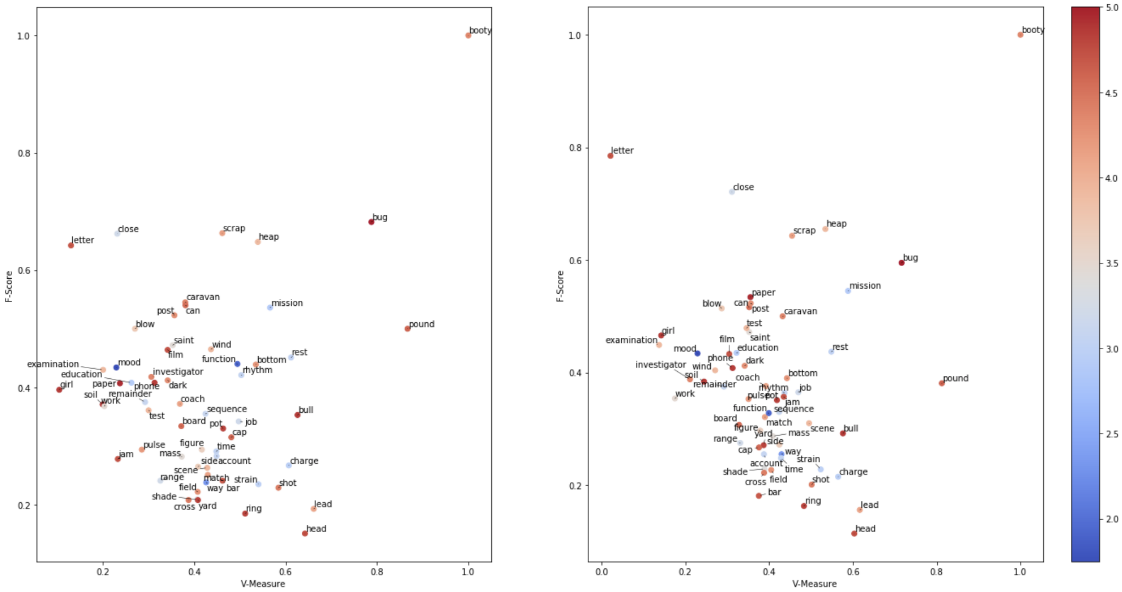
Figure 41: Gold clustering along with the predicted clusterings for the word ‘fundamental.a’

Adjectives Figure 41 contains an example of clusterings for the adjective 'fundamental.a'. The gold clustering contains 3 distinct senses for the target word, which can perhaps be summarised as central, profound, and rudimentary. The performance of image features as supposed to word2vec is significantly worse. The difference between predicted clusterings produced by image atures and word2vec is that the paraphrase 'primal' is removed and made as a separate cluster. Such move penalises image features in both V-Measure and F-Score, since 'primal' had the word 'cardinal' that belonged together in the original green cluster. Such change has 0.15 drop in performance in both of the evaluation metrics. None of the configurations seem to misplace 'profound', 'underlying', and 'rudimentary', but more importantly, separate 'primal' and/or 'cardinal' with 'central, key'.

4.6.6. Concreteness

The concreteness of a word can also play a helpful role in incorporating features into clustering tasks. The concreteness of a target word is inversely proportional to the degree of vagueness of its meaning. For example, 'strawberry' is highly concrete, but 'job' is relatively less concrete, since it can assume multiple meanings. The intuition behind this metric is that if the concreteness of the target word is high, the more likely it is that the image features will be highly complementary to the clustering task. In order to evaluate the performance by concreteness for each target word we extracted a concreteness score based on the values from Brysbaert et al. (2013). For WordNet+ Gold 2.0 dataset the minimum concreteness score is 1.19, maximum is 5.0, and the mean at 3.17. The higher the score, the more concrete the word is. For instance, the word 'mood' has 1.75 set as the concreteness score, but the word 'soil' has a score of 4.87. In order to perform the analysis, we separated the dataset by POS and decided to plot F-Score against V-Measure with concreteness score visualised by the colour. Blue displays a low concreteness score, whereas red displays a high concreteness score.

Nouns For nouns the minimum concreteness score is 1.75, maximum is 5 and mean is 3.93. Figure 42 displays the performance of two configurations, where on the left images are set as the input to the similarity matrix and on the right word2vec is set as the input to the similarity matrix. For nouns the F-Score and V-Measure in both configurations are clustered together in the lower left corner with colours mixed. For both image features and word2vec a clear pattern cannot be observed for concreteness.

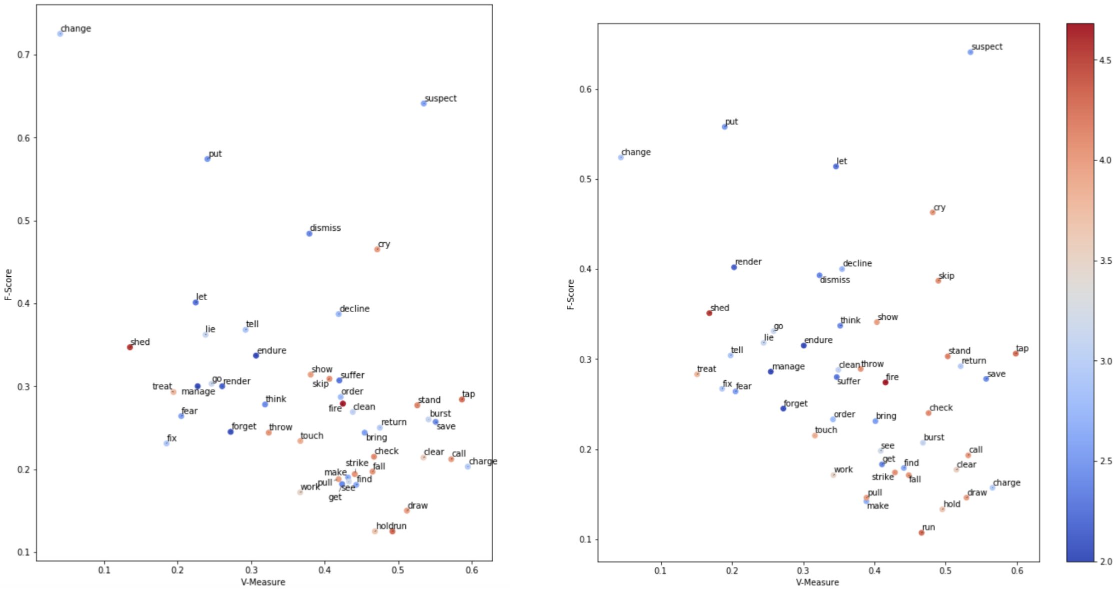


(a) Similarity Matrix = Images, Static K

(b) Similarity Matrix = Word2vec, Static K

Figure 42: F-Score vs V-Measure plot of predicted scores for a target word coloured by concreteness for nouns from WordNet+ Gold 2.0 dataset

Verbs For verbs the minimum concreteness score is 2, maximum is 4.68 and mean is 3.21. Figure 43 displays the performance of image features and word2vec configurations on verbs. Unlike nouns, for verbs F-Score and V-Measure in both configurations are more distributed, with V-Measure reaching higher values than the F-Score on average. For image features, a pattern can be observed, where there values that fall under F-Score of less than 0.4 and V-Measure of less than 0.4 have low concreteness scores. A similar observation, but a smaller window size can be be observed for word2vec.

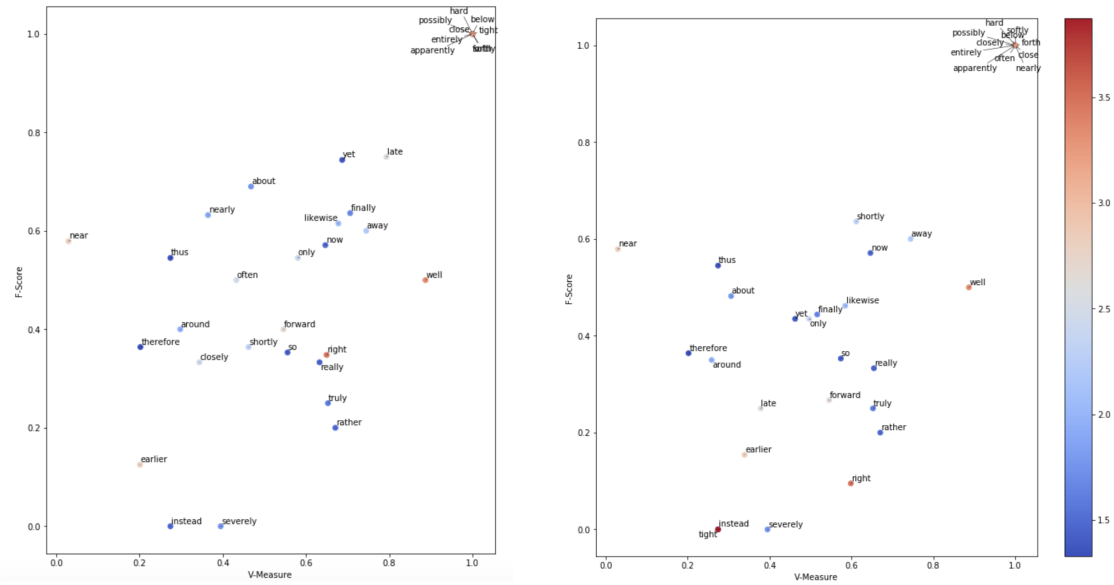


(a) Similarity Matrix = Images, Static K

(b) Similarity Matrix = Word2vec, Static K

Figure 43: F-Score vs V-Measure plot of predicted scores for a target word coloured by concreteness for verbs from WordNet+ Gold 2.0 dataset

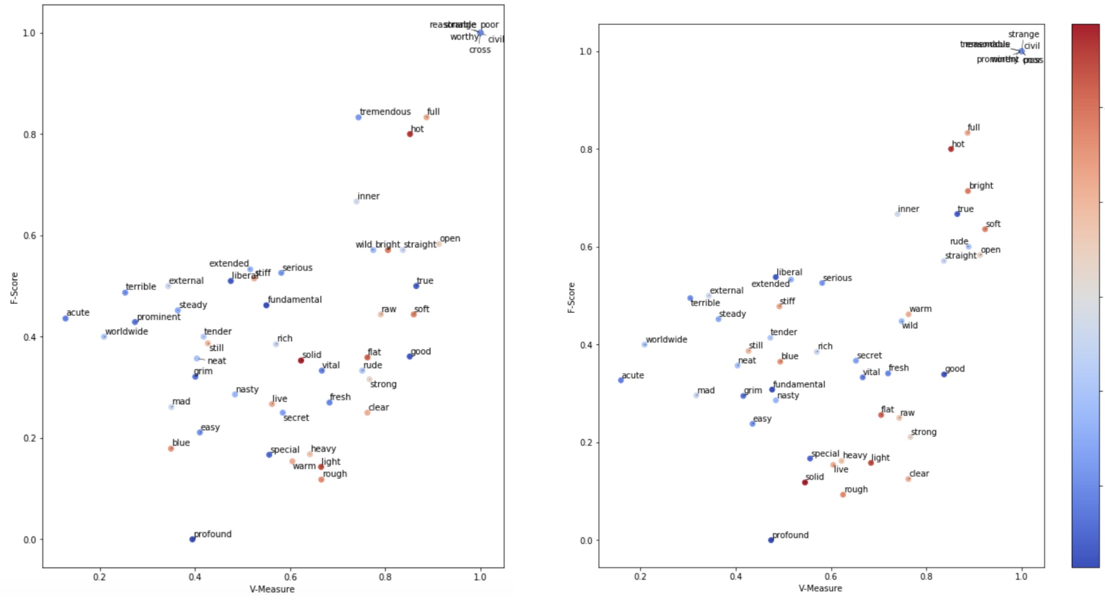
Adverbs For adverbs the minimum concreteness score is 1.33, maximum is 3.87 and mean is 2.25. Figure 44 displays the performance of image features and word2vec configurations on adverbs. Similarly to verbs, the points in the plot for both configurations are very sparse. It can be seen that majority of the words have a low concreteness score, hence there colour of majority of points are blue. In the right upper corner words that achieved the best possible performance in terms of F-Score and V-Measure can be seen. Some of the words include ‘hard’, ‘possibly’, ‘entirely’, ‘apparently’ and so on.



(a) Similarity Matrix = Images, Static K

(b) Similarity Matrix = Word2vec, Static K

Figure 44: F-Score vs V-Measure plot of predicted scores for a target word coloured by concreteness for adverbs from WordNet+ Gold 2.0 dataset



(a) Similarity Matrix = Images, Static K

(b) Similarity Matrix = Word2vec, Static K

Figure 45: F-Score vs V-Measure plot of predicted scores for a target word coloured by concreteness for adjectives from WordNet+ Gold 2.0 dataset

Adjectives For adjectives the minimum concreteness score is 1.57, maximum is 4.44 and mean is 2.74. Figure 45 displays the performance of image features and word2vec configurations on adjectives. The distribution of points on the plot for adjectives resemble the distribution of verbs. What can be observed from both of the configurations is on average, the more concrete the target word is, the higher is the V-Measure score is for adjectives. Similarly to adverbs, the right upper corner words that achieved the best possible performance in terms of F-Score and V-Measure can be seen. Some of the words include ‘worthy’, ‘poor’, ‘strange’, and ‘civil’.

CHAPTER 5 : Discussion and Future Work

For the task of word similarity prediction, certain observations can be made based on the results seen in Chapter 3. To start with, even though thirteen datasets were used to evaluate the performance of different vector representations, only a small subset of datasets provide a sufficient number of data points with reliable human judgement scores. As a result, we prioritised datasets such as SimLex-999 in our quantitative and qualitative analysis. Based on the results on SimLex-999 dataset, as the number of image features increases, the performance increases for all three visual representations. This is especially evident by the performance on 666 nouns and 111 adjectives from SimLex-999 dataset. A similar pattern can be observed on MEN-3000, MTurk-287, and MTurk-771 datasets. On the other hand, as the number of image features increases the performance decreases for 222 verbs from SimLex-999 dataset. In general, all three visual representations have a significantly worse performance on datasets that consist only with verbs, for instance Verb-143, SimVerb-3500, and 222 verbs from SimLex-999 datasets. This suggests that part-of-speech should play a role when choosing to use visual representation. While the performance of linguistic representation is better for other datasets, visual representation is comparable to linguistic on SimLex-999 dataset. This suggests that image-based representation can be used as an alternative way of predicting similarity of word pairs. We have tried combining visual and textual representations through concatenation of vectors, however the performance of this multimodal representation was worse as supposed to using visual or lingustic features in isolation. Further comparison of two representations done as part of qualitative analysis revealed that those unimodal representations predict certain word pairs very similarly. However, the range of values predicted using AvgMax approach is much more narrow in comparison to the linguistic approach. Such observation needs further investigation and can be done as part of future work.

For the task of clustering paraphrases by word sense seen in Chapter 4, many conclusions can be made based on the results. First, most clustering algorithms require an affinity or similarity matrix as an input, which denotes a pairwise similarity between paraphrases. This thesis explored are a variety of choices for populating a similarity matrix, for example linguistic, image-based or contextual (from images) representations. In addition to running existing similarity measures in isolation, there are numerous way of combining those unimodal representations to form a multimodal representation. This thesis explored one of the simplest ways of combining representations by averaging a similarity score between a pair of paraphrases from various modes and then performing normalisation. The results have shown that a multimodal representation of textual and visual representations work best on CrowdClusters dataset, but does not work so well for the WordNet+ dataset. Another way to combine different modes of data would have been to assign weights for a particular representation based on a target word or its properties, such as part-of-speech. This approach of combining was not explored in the thesis, but could be done in future. Going forward from our approach of combining the similarities of embeddings before clustering, another extension would be to explore an ensembling technique to combine the clusters obtained by each of the features independently. To paraphrase, a clustering algorithm would run on different unimodal representations in isolation and then an ensembling happens post-clustering.

The reason why such approaches were not explored in the thesis was due to time constraints, but also because the evaluation of different models resulted in similar performances on WordNet+ and CrowdClusters datasets. We believe that the existing datasets suffer from a low quality of clusters due to various reasons: noisiness, over 90% of paraphrases in one cluster, depiction of outdated senses, foreign characters, informal words and so on. Moreover, the input file and gold file do not have sufficient overlap between the two sets of paraphrases for a given target word, making the quality of clusters even lower. These limitations served as a motivation for creating a new dataset from the gold file of WordNet+ dataset and removing certain number of paraphrases that do not appear in the collection of Callahan (2017). The results on a new dataset showed that visual representation and textual representation are comparable to each other in terms of performance. In addition, we observed that the performance of image features in comparison to textual features is higher for nouns and adverbs and is lower for verbs and adjectives. This can be explained by the inherent nature of the images.

Overall, we believe that the idea of linking a query word to a set of images from the search engine is very powerful, since in theory, sets of images can be extracted for any query word or phrase, no matter how complex a query word is. One obvious extension from this thesis would be to use a different neural network architecture when converting a set of images to a set of image features. This could mean varying the number of dimensions created for each image features, perhaps reducing the size of dimensions to be comparable to dense textual representations. For both of the explored tasks, there are numerous ways of combining different modes of data into a multimodal representation. Perhaps there is a need to think of alternative ways of combining text and images. Finally, one of the most noticeable observations was that visual features suffer from representing verbs. There should be a consideration on how to treat verbs, perhaps using videos instead of image features as a representation.

APPENDIX

Dataset	No Images						
	Top-1	Top-5	Top-10	Top-25	Top-50	Top-75	Top-100
RG-65	0.48	0.4658	0.5105	0.56	0.6295	0.6397	0.6422
MC-30	0.4041	0.4262	0.4251	0.4903	0.5848	0.5726	0.5978
WordSimilarity-353-ALL	0.2295	0.2985	0.2726	0.2583	0.2473	0.2401	0.2366
WordSimilarity-353-SIM	0.3638	0.3579	0.3601	0.3442	0.3424	0.3311	0.3267
WordSimilarity-353-REL	0.108	0.1877	0.1397	0.1275	0.109	0.1046	0.1016
MTurk-287	0.2088	0.3073	0.3144	0.3171	0.31	0.3115	0.3042
MTurk-771	0.275	0.3649	0.3848	0.4007	0.4199	0.4193	0.4153
MEN-3000	0.4394	0.5206	0.5297	0.5331	0.5415	0.54	0.5405
SimLex-999	0.2271	0.3127	0.3367	0.3596	0.3795	0.3789	0.376
SimLex-666-Nouns	0.2753	0.3586	0.3974	0.4273	0.4682	0.4773	0.4774
SimLex-222-Verbs	0.2004	0.2103	0.1731	0.1747	0.1406	0.107	0.0898
SimLex-111-Adjectives	0.0217	0.2095	0.27	0.3403	0.3492	0.3687	0.3721
YP-130	0.0801	0.0996	0.0989	0.0951	0.1208	0.1308	0.1325
VERB-143	-0.0075	0.0324	-0.0227	-0.0755	-0.0503	-0.0417	-0.0323
SimVerb-3500	0.0603	0.0967	0.1066	0.1085	0.1046	0.0995	0.094
RW	0.0819	0.238	0.251	0.2849	0.2785	0.2809	0.2854

Table 31: Performance of Avg(w1, w2) on predicting word similarity. (Section 2.3.1)

Dataset	No Images						
	Top-2	Top-5	Top-10	Top-25	Top-50	Top-75	Top-100
RG-65	0.4691	0.4892	0.5575	0.5647	0.5785	0.5901	0.55
MC-30	0.332	0.4564	0.4829	0.4818	0.4535	0.442	0.373
WordSimilarity-353-ALL	0.2886	0.3462	0.316	0.2859	0.3006	0.2984	0.2807
WordSimilarity-353-SIM	0.3721	0.4216	0.4121	0.4036	0.4146	0.4187	0.395
WordSimilarity-353-REL	0.1629	0.2256	0.1883	0.1413	0.1741	0.1537	0.132
MTurk-287	0.3237	0.3266	0.3308	0.3618	0.377	0.3883	0.3754
MTurk-771	0.3167	0.3607	0.401	0.4231	0.4466	0.4467	0.4401
MEN-3000	0.5173	0.5352	0.551	0.5754	0.5928	0.5974	0.6032
SimLex-999	0.2666	0.2959	0.3231	0.3342	0.3566	0.3527	0.3464
SimLex-666-Nouns	0.3186	0.3537	0.3873	0.4041	0.4457	0.4474	0.4417
SimLex-222-Verbs	0.212	0.1786	0.1586	0.1427	0.1084	0.0953	0.0887
SimLex-111-Adjectives	0.0731	0.1862	0.2292	0.2549	0.249	0.2603	0.2434
YP-130	0.1022	0.1245	0.1069	0.0918	0.1534	0.1475	0.1782
VERB-143	-0.0677	-0.0041	-0.0101	-0.0607	-0.0771	-0.0322	-0.0039
SimVerb-3500	0.0859	0.0936	0.108	0.1086	0.1056	0.0991	0.0958
RW	0.1772	0.2108	0.2116	0.2558	0.2414	0.2295	0.2267

Table 32: Performance of AvgMax on predicting word similarity. (Section 2.3.2)

Dataset	No Images						
	Top-2	Top-5	Top-10	Top-25	Top-50	Top-75	Top-100
RG-65	0.4496	0.4521	0.4646	0.4724	0.5005	0.5255	0.534
MC-30	0.3145	0.3966	0.3607	0.3574	0.3765	0.401	0.4093
WordSimilarity-353-ALL	0.2913	0.3255	0.3022	0.2926	0.2931	0.3023	0.2964

WordSimilarity-353-SIM	0.3818	0.4127	0.4109	0.4081	0.4158	0.4241	0.42
WordSimilarity-353-REL	0.1645	0.2097	0.1707	0.163	0.1632	0.1695	0.1621
MTurk-287	0.3233	0.3366	0.3387	0.3704	0.3891	0.3973	0.3897
MTurk-771	0.3382	0.3615	0.393	0.4088	0.4264	0.4248	0.4215
MEN-3000	0.5292	0.5468	0.5625	0.5833	0.6006	0.6065	0.6094
SimLex-999	0.2548	0.2603	0.2667	0.2726	0.2875	0.2838	0.2797
SimLex-666-Nouns	0.3089	0.322	0.3314	0.3347	0.361	0.3667	0.3641
SimLex-222-Verbs	0.2094	0.1854	0.134	0.141	0.1222	0.0765	0.0519
SimLex-111-Adjectives	0.0278	0.0693	0.1186	0.1404	0.1496	0.1282	0.1391
YP-130	0.0962	0.0962	0.0881	0.1008	0.1501	0.1664	0.1917
VERB-143	-0.0545	-0.0138	-0.0445	-0.0911	-0.0914	-0.0762	-0.0698
SimVerb-3500	0.0725	0.0726	0.0759	0.0781	0.0718	0.0655	0.0628
RW	0.1566	0.1835	0.1832	0.1953	0.1904	0.1836	0.1815

Table 33: Performance of AvgAvg on predicting word similarity. (Section 2.3.3)

Dataset	**No Images**					
	First 100	First 200	First 300	First 1000	First 2000	All 4096 points
RG-65	0.6877	0.6713	0.6415	0.644	0.6439	0.6422
MC-30	0.6727	0.7008	0.6158	0.6055	0.6026	0.5978
WordSimilarity-353-ALL	0.2782	0.2654	0.2652	0.2541	0.2422	0.2366
WordSimilarity-353-SIM	0.3405	0.3495	0.3459	0.3382	0.3264	0.3267
WordSimilarity-353-REL	0.1611	0.1358	0.1334	0.1227	0.1114	0.1016
MTurk-287	0.2842	0.287	0.3107	0.3055	0.3056	0.3042
MTurk-771	0.4168	0.4176	0.4133	0.4153	0.4152	0.4153
MEN-3000	0.4928	0.5123	0.5216	0.5345	0.5355	0.5405
SimLex-999	0.3583	0.3688	0.3718	0.3787	0.3766	0.376
SimLex-666-Nouns	0.4696	0.4767	0.4711	0.4807	0.4803	0.4774
SimLex-222-Verbs	0.0206	0.0558	0.0787	0.0867	0.0901	0.0898
SimLex-111-Adjectives	0.3342	0.3304	0.3465	0.3663	0.3688	0.3721
YP-130	0.1072	0.1145	0.1207	0.1327	0.1304	0.1325
VERB-143	0.0142	0.0142	0.0183	-0.0276	-0.0335	-0.0323
SimVerb-3500	0.0903	0.092	0.0921	0.0926	0.0934	0.094
RW	0.2714	0.2875	0.2901	0.2869	0.2841	0.2854

Table 34: Performance of Avg(w1, w2) For Top-100 images varying the number of dimensions

Dataset	**PCA to 100 averaged**	**PCA to 300 averaged**	**Word2Vec + First 300**	**Word2Vec + PCA to 300 averaged**
RG-65	0.5447	0.5476	0.6422	0.5847
MC-30	0.4818	0.4869	0.6222	0.535
WordSimilarity-353-ALL	0.3936	0.3959	0.2702	0.1539
WordSimilarity-353-SIM	0.5106	0.5155	0.3529	0.2098
WordSimilarity-353-REL	0.2622	0.2605	0.1367	0.0404
MTurk-287	0.469	0.4748	0.3181	0.2173
MTurk-771	0.4649	0.4686	0.4184	0.3154
MEN-3000	0.6087	0.6089	0.5262	0.2339
SimLex-999	0.2892	0.2952	0.3756	0.3392
SimLex-666-Nouns	0.3806	0.3878	0.4731	0.3825
SimLex-222-Verbs	0.0757	0.0772	0.0848	0.2495

SimLex-111-Adjectives	0.2867	0.2914	0.3531	0.5341
YP-130	0.2065	0.2092	0.126	0.1969
VERB-143	0.0382	0.0324	0.0225	0.1277
SimVerb-3500	0.1221	0.1229	0.0964	0.1801
RW	0.2763	0.279	0.296	0.3792

Table 35: Performance of additional models on predicting word similarity

BIBLIOGRAPHY

Cocos, A.; Callison-Burch, C. Clustering paraphrases by word sense. Proceedings of the 2016 Conference of the North American Chapter of the Association for Computational Linguistics: Human Language Technologies. 2016; pp 1463–1472.

Callahan, B. D. Image-based Bilingual Lexicon Induction for Low Resource Languages. 2017.

Bergsma, S.; Van Durme, B. Learning Bilingual Lexicons Using the Visual Similarity of Labeled Web Images. Proceedings of the Twenty-Second International Joint Conference on Artificial Intelligence - Volume Volume Three. 2011; pp 1764–1769.

Hewitt, J.; Ippolito, D.; Callahan, B.; Kriz, R.; Wijaya, D.; Callison-Burch, C. Learning Translations via Images with a Massively Multilingual Image Dataset. Association for Computational Linguistics. 2018.

Miller, G. A. *Commun. ACM* **1995**, *38*, 39–41.

Harris, Z. *Word* **1954**, *10*, 146–162.

Brysbaert, M.; Beth Warriner, A.; Kuperman, V. *Behavior research methods* **2013**, *46*.

Kiela, D.; Verő, A. L.; Clark, S. Comparing Data Sources and Architectures for Deep Visual Representation Learning in Semantics. Proceedings of the 2016 Conference on Empirical Methods in Natural Language Processing. Austin, Texas, 2016; pp 447–456.

Kiela, D.; Vulić, I.; Clark, S. Visual Bilingual Lexicon Induction with Transferred ConvNet Features. Proceedings of the 2015 Conference on Empirical Methods in Natural Language Processing. Lisbon, Portugal, 2015; pp 148–158.

Krizhevsky, A.; Sutskever, I.; Hinton, G. E. Imagenet classification with deep convolutional neural networks. Advances in neural information processing systems. 2012; pp 1097–1105.

Bullinaria, J. A.; Levy, J. P. *Behavior Research Methods* **2007**, *39*, 510–526.

Padó, S.; Lapata, M. *Comput. Linguist.* **2007**, *33*, 161–199.

Kiela, D.; Bottou, L. Learning Image Embeddings using Convolutional Neural Networks for Improved Multi-Modal Semantics. Proceedings of the 2014 Conference on Empirical Methods in Natural Language Processing (EMNLP). Doha, Qatar, 2014; pp 36–45.

Bergsma, S.; Goebel, R. Using Visual Information to Predict Lexical Preference. Proceedings of the International Conference Recent Advances in Natural Language Processing 2011. Hissar, Bulgaria, 2011; pp 399–405.

Bruni, E.; Tran, N. K.; Baroni, M. *J. Artif. Int. Res.* **2014**, *49*, 1–47.

von Ahn, L.; Dabbish, L. Labeling Images with a Computer Game. Proceedings of the SIGCHI Conference on Human Factors in Computing Systems. New York, NY, USA, 2004; pp 319–326.

Sivic, J.; Zisserman, A. Video Google: A Text Retrieval Approach to Object Matching in Videos. Proceedings of the Ninth IEEE International Conference on Computer Vision - Volume 2. Washington, DC, USA, 2003; pp 1470–.

Kiela, D.; Hill, F.; Korhonen, A.; Clark, S. Improving Multi-Modal Representations Using Image Dispersion: Why Less is Sometimes More. Proceedings of the 52nd Annual Meeting of the Association for Computational Linguistics (Volume 2: Short Papers). Baltimore, Maryland, 2014; pp 835–841.

Kiela, D.; Rimell, L.; Vulić, I.; Clark, S. Exploiting Image Generality for Lexical Entailment Detection. Proceedings of the 53rd Annual Meeting of the Association for Computational Linguistics and the 7th International Joint Conference on Natural Language Processing (Volume 2: Short Papers). Beijing, China, 2015; pp 119–124.

Shutova, E.; Kiela, D.; Maillard, J. Black Holes and White Rabbits: Metaphor Identification with Visual Features. Proceedings of the 2016 Conference of the North American Chapter of the Association for Computational Linguistics: Human Language Technologies. San Diego, California, 2016; pp 160–170.

Bulat, L.; Kiela, D.; Clark, S. Vision and Feature Norms: automatic feature norm learning through cross-modal maps. Proceedings of the 2016 Conference of the North American Chapter of the Association for Computational Linguistics: Human Language Technologies. San Diego, California, 2016; pp 579–588.

Vulić, I.; Kiela, D.; Clark, S.; Moens, M.-F. Multi-Modal Representations for Improved Bilingual Lexicon Learning. Proceedings of the 54th Annual Meeting of the Association for Computational Linguistics (Volume 2: Short Papers). Berlin, Germany, 2016; pp 188–194.

Lazaridou, A.; Pham, N. T.; Baroni, M. Combining Language and Vision with a Multimodal Skip-gram Model. Proceedings of the 2015 Conference of the North American Chapter of the Association for Computational Linguistics: Human Language Technologies. Denver, Colorado, 2015; pp 153–163.

Mikolov, T.; Sutskever, I.; Chen, K.; Corrado, G.; Dean, J. Distributed Representations of Words and Phrases and Their Compositionality. Proceedings of the 26th International Conference on Neural Information Processing Systems. 2013; pp 3111–3119.

Kiela, D. MMFeat: A Toolkit for Extracting Multi-Modal Features. Proceedings of ACL-2016 System Demonstrations. Berlin, Germany, 2016; pp 55–60.

Anderson, A. J.; Kiela, D.; Clark, S.; Poesio, M. *Transactions of the Association for Computational Linguistics* **2017**, *5*, 17–30.

Bulat, L.; Clark, S.; Shutova, E. Speaking, Seeing, Understanding: Correlating semantic models with conceptual representation in the brain. Proceedings of the 2017 Conference on Empirical Methods in Natural Language Processing. Copenhagen, Denmark, 2017; pp 1081–1091.

Glavaš, G.; Vulić, I.; Ponzetto, S. P. If Sentences Could See: Investigating Visual Information for Semantic Textual Similarity. IWCS 2017 - 12th International Conference on Computational Semantics - Long papers. 2017.

Simonyan, K.; Zisserman, A. *CoRR* **2014**, *abs/1409.1556*.

Russakovsky, O.; Deng, J.; Su, H.; Krause, J.; Satheesh, S.; Ma, S.; Huang, Z.; Karpathy, A.; Khosla, A.; Bernstein, M.; et al., *International Journal of Computer Vision* **2015**, *115*, 211252.

Bhaskar, S. A.; Köper, M.; Schulte im Walde, S.; Frassinelli, D. Exploring Multi-Modal Text+Image Models to Distinguish between Abstract and Concrete Nouns. Proceedings of the IWCS workshop on Foundations of Situated and Multimodal Communication. 2017.

Hartmann, M.; Søgaard, A. Limitations of Cross-Lingual Learning from Image Search. Proceedings of The Third Workshop on Representation Learning for NLP. Melbourne, Australia, 2018; pp 159–163.

Wang, S.; Zhang, J.; Zong, C. Learning Multimodal Word Representation via Dynamic Fusion Methods. Proceedings of the Thirty-Second AAAI Conference on Artificial Intelligence, (AAAI-18), the 30th innovative Applications of Artificial Intelligence (IAAI-18), and the 8th AAAI Symposium on Educational Advances in Artificial Intelligence (EAAI-18), New Orleans, Louisiana, USA, February 2-7, 2018. 2018; pp 5973–5980.

Collell, G.; Zhang, T.; Moens, M. Imagined Visual Representations as Multimodal Embeddings. Proceedings of the Thirty-First AAAI Conference on Artificial Intelligence, February 4-9, 2017, San Francisco, California, USA. 2017; pp 4378–4384.

Collell, G.; Moens, M.-F. Do Neural Network Cross-Modal Mappings Really Bridge Modalities? Proceedings of the 56th Annual Meeting of the Association for Computational Linguistics (Volume 2: Short Papers). Melbourne, Australia, 2018; pp 462–468.

Kiros, J.; Chan, W.; Hinton, G. Illustrative Language Understanding: Large-Scale Visual Grounding with Image Search. Proceedings of the 56th Annual Meeting of the Association for Computational Linguistics (Volume 1: Long Papers). Melbourne, Australia, 2018; pp 922–933.

Pennington, J.; Socher, R.; Manning, C. Glove: Global Vectors for Word Representation. Proceedings of the 2014 Conference on Empirical Methods in Natural Language Processing (EMNLP). Doha, Qatar, 2014; pp 1532–1543.

Turney, P. D.; Pantel, P. *J. Artif. Int. Res.* **2010**, *37*, 141–188.

Clark, S. *Handbook of Contemporary Semantics* **2015**,

Mikolov, T.; Yih, W.-t.; Zweig, G. Linguistic Regularities in Continuous Space Word Representations. Proceedings of the 2013 Conference of the North American Chapter of the Association for Computational Linguistics: Human Language Technologies. Atlanta, Georgia, 2013; pp 746–751.

Joulin, A.; Grave, E.; Bojanowski, P.; Douze, M.; Jégou, H.; Mikolov, T. *CoRR* **2016**, *abs/1612.03651*.

Peters, M. E.; Neumann, M.; Iyyer, M.; Gardner, M.; Clark, C.; Lee, K.; Zettlemoyer, L. Deep contextualized word representations. Proc. of NAACL. 2018.

Griffiths, T. L.; Tenenbaum, J. B.; Steyvers, M. *Psychological Review* **2007**, *114*, 2007.

Finkelstein, L.; Gabrilovich, E.; Matias, Y.; Rivlin, E.; Solan, Z.; Wolfman, G.; Ruppin, E. Placing Search in Context: The Concept Revisited. Proceedings of the 10th International Conference on World Wide Web. New York, NY, USA, 2001; pp 406–414.

Lund, K.; Burgess, C. *Behavior Research Methods Instruments and Computers* **1996**, *28*, 203–208.

Golub, G. H.; Reinsch, C. *Numer. Math.* **1970**, *14*, 403–420.

Silberer, C.; Lapata, M. Learning Grounded Meaning Representations with Autoencoders. Proceedings of the 52nd Annual Meeting of the Association for Computational Linguistics (Volume 1: Long Papers). Baltimore, Maryland, 2014; pp 721–732.

Hill, F.; Reichart, R.; Korhonen, A. *Computational Linguistics* **2015**, *41*, 665–695.

Szegedy, C.; Liu, W.; Jia, Y.; Sermanet, P.; Reed, S.; Anguelov, D.; Erhan, D.; Vanhoucke, V.; Rabinovich, A. Going Deeper with Convolutions. Computer Vision and Pattern Recognition (CVPR). 2015.

Agirre, E.; Alfonseca, E.; Hall, K.; Kravalova, J.; Paşca, M.; Soroa, A. A Study on Similarity and Relatedness Using Distributional and WordNet-based Approaches. Proceedings of Human Language Technologies: The

2009 Annual Conference of the North American Chapter of the Association for Computational Linguistics. Stroudsburg, PA, USA, 2009; pp 19–27.

Rubenstein, H.; Goodenough, J. B. *Commun. ACM* **1965**, *8*, 627–633.

Miller, G. A.; Charles, W. G. *Language and Cognitive Processes* **1991**, *6*, 1–28.

Radinsky, K.; Agichtein, E.; Gabrilovich, E.; Markovitch, S. A Word at a Time: Computing Word Relatedness Using Temporal Semantic Analysis. Proceedings of the 20th International Conference on World Wide Web. New York, NY, USA, 2011; pp 337–346.

Halawi, G.; Dror, G.; Gabrilovich, E.; Koren, Y. Large-scale Learning of Word Relatedness with Constraints. Proceedings of the 18th ACM SIGKDD International Conference on Knowledge Discovery and Data Mining. New York, NY, USA, 2012; pp 1406–1414.

Resnik, P.; Lin, J. 11. Evaluations of NLP systems. The handbook of computational linguistics and natural language processing. 2010; pp 271–295.

Yang, D.; Powers, D. Verb similarity on the taxonomy of WordNet - dataset. 3rd International WordNet Conference (GWC-06). 2006.

Baker, S.; Reichart, R.; Korhonen, A. An Unsupervised Model for Instance Level Subcategorization Acquisition. Proceedings of the 2014 Conference on Empirical Methods in Natural Language Processing (EMNLP). Doha, Qatar, 2014; pp 278–289.

Gerz, D.; Vulić, I.; Hill, F.; Reichart, R.; Korhonen, A. SimVerb-3500: A Large-Scale Evaluation Set of Verb Similarity. Proceedings of the 2016 Conference on Empirical Methods in Natural Language Processing. Austin, Texas, 2016; pp 2173–2182.

Luong, T.; Socher, R.; Manning, C. Better Word Representations with Recursive Neural Networks for Morphology. Proceedings of the Seventeenth Conference on Computational Natural Language Learning. Sofia, Bulgaria, 2013; pp 104–113.

Faruqui, M.; Dyer, C. Community Evaluation and Exchange of Word Vectors at wordvectors.org. Proceedings of ACL: System Demonstrations. 2014.

Bannard, C.; Callison-Burch, C. Paraphrasing with Bilingual Parallel Corpora. Proceedings of the 43rd Annual Meeting on Association for Computational Linguistics. Stroudsburg, PA, USA, 2005; pp 597–604.

Callison-Burch, C. Syntactic constraints on paraphrases extracted from parallel corpora. Proceedings of the Conference on Empirical Methods in Natural Language Processing. 2008; pp 196–205.

Apidianaki, M.; Verzeni, E.; McCarthy, D. Semantic Clustering of Pivot Paraphrases. LREC. 2014; pp 4270–4275.

Pavlick, E.; Rastogi, P.; Ganitkevitch, J.; Van Durme, B.; Callison-Burch, C. PPDB 2.0: Better paraphrase ranking, fine-grained entailment relations, word embeddings, and style classification. 2015; pp 425–430.

Zelnik-Manor, L.; Perona, P. Self-tuning Spectral Clustering. Proceedings of the 17th International Conference on Neural Information Processing Systems. Cambridge, MA, USA, 2004; pp 1601–1608.

Rousseeuw, P. J. *Journal of computational and applied mathematics* **1987**, *20*, 53–65.

Manandhar, S.; Klapaftis, I. P.; Dligach, D.; Pradhan, S. S. SemEval-2010 Task 14: Word Sense Induction & Disambiguation. Proceedings of the 5th International Workshop on Semantic Evaluation. Stroudsburg, PA, USA, 2010; pp 63–68.