technische universität
dortmund

Bachelor's Thesis

Mapping Co-Occurrence Vectors Built upon Large Text Corpora to a Human-Interpretable Space Using Explainable Decision Trees

Sebastian Schmidt
March 2021

Supervisors:
Prof. Dr. Katharina Morik
Lukas Pfahler (M.Sc.)

Technische Universität Dortmund
Fakultät für Informatik
Lehrstuhl für Künstliche Intelligenz (LS8)
https://www-ai.cs.tu-dortmund.de/index.html

Contents

Chapter 1

Introduction

1.1 Motivation and Background

One permanently growing field in modern AI research is *Natural Language Processing* (NLP). It aims for models which contain some kind of knowledge about processing texts written in plain natural language, rather than a formalized representation; these include spam filters, semantic based search engines, speech recognition and sentiment analysis, among others. For some of those applications, so-called *word embeddings* – of which word2vec [20] is one of the most popular – have been developed: These introduce methods to embed words in a vector space where the distance and direction of word vectors to each other should encode parts of their meaning.

The whole idea grounds on the distributional hypothesis of Harris [12], stating that meaning and grammatical function of words can be described by observing the words they usually appear with. In the case of grammar, that may be articles, rendering following words as nouns, or "to" which in specific frequencies leads us to suspect the following word to be an infinitive and as such a verb. In the case of semantics, "Paris" and "Berlin" would surely appear more often close to "capital" than most words.

This kind of relation is usually represented in a co-occurrence matrix, where every row and column represents a specific word of the observed corpus and every cell the evidence that each two words share a strong relation, i.e. commonly appear in the same context. The cell may just contain an absolute number of counts, which, however, would not take into account the generally lower number of rare words. Instead, usually a metric like the pointwise mutual information (PMI) is used.

There are some problems with these co-occurrence matrices, nonetheless: they are not in any form expressive and require a lot of memory for large text corpora. As a consequence, they are not particularly helpful in most real-world uses. With word embeddings calculated

via neural networks, the last and partly the first point are confronted: in principle the method can be described by a matrix factorization on a shifted co-occurrence matrix [17] which reduces the overall dimension whilst preserving the most important inherent properties. However, the algorithm avoids collecting the complete co-occurrence matrix by training a neural network on doing a similar job: mapping a word to its context or vice versa predicting the context from a word. The corresponding network's weights are then used as a word vector.

Another approach determines the factors of a linear mapping to construct vectors according to a few examples [8].

However, both of these methods lack *explainability* [7, § A]: The mappings themselves are complex and hard to understand for humans, i.e. why exactly two words achieve a high similarity or are recognized as a specific category. Additionally, typical neural-network-based word embeddings do not grant a human observer any obvious information and some work is needed to find suitable subspaces for given semantic properties.

It is stated in [4], that this might be quite problematic depending on its field of use:

> "There have been hundreds or thousands of papers written about word embeddings and their applications, from Web search to parsing Curriculum Vitae. However, none of these papers have recognized how blatantly sexist the embeddings are and hence risk introducing biases of various types into real-world systems." [4]

Since word embeddings are usually trained unsupervised and need very large chunks of text to be most accurate, so large that no developer can read through them beforehand to cut out potentially problematic content, they are at risk to inadvertently learn negative or stereotypical associations being wide-spread throughout the corpus. The opaque vector construction as well as its unknown vector space structure make it very difficult to recognize these problems, especially if not manually searching for them, and even harder to avoid them.

This paper explores a novel approach which also seeks to construct a reduced vector space based on single words, only that this vector space is manually designed so that each component describes the membership to a specific human-understandable category. The idea itself is not new, it has already been proposed in [14] and [8] several years ago.

However, the transformation evaluated in the following grounds on *decision trees* and follows an implementation such that for every sample a justification for its categorization can be given. It should also be possible to notice problems simply by taking a look at the tree structure: if the word "man" is used as a positively discriminating feature for professions, action should be taken to avoid this consequence.

1.2 Structure of the Thesis

Before the actual work of this thesis will be explored, some basic knowledge and definitions are to be provided. This includes the necessary explanations to the distributional hypothesis and co-occurence matrices, the pointwise mutual information metric, word embeddings as well as decision trees. That is the role dedicated to Chapter 2.

In Chapter 3, the idea behind this thesis is elaborated in more detail and two manually designed target spaces used for testing are presented.

Chapter 4 offers a short introduction into the Python implementation of the presented approach which relies on the Scikit-learn python package [25] for the most part.

Chapter 5 is then dedicated to the evaluation, and also lists some problems one may confront in the process of implementation and testing.

Finally, in Chapter 6, the evaluation results are summarized in a short conclusion which decides whether the proposed approach is actually applicable in practice, or where it could be applicable.

Chapter 2

Definitions & Basics

2.1 The Distributional Hypothesis

The distributional hypothesis came up in the mid-20th century, with [12] being one of the first papers covering this topic. The main assumption of this hypothesis is that expressions in natural languages are mainly used in specific contexts – expressions could therefore be described by the set of all contexts they appear in. To clarify the idea, a few examples:

- Words appearing very often on the left side of the word "eat", i.e. as the subject, can be presumed as describing living beings. On the other hand, terms representing abstract objects almost never occur in this position.
- From a grammatical point of view, the German verb form "hast" (second person singular of "haben", *to have*) will most of the time occur in a sentence with "du" (Germ., *you*), as well will other verbs conjugated accordingly.
- Regarding topics, for a word occurring frequently in paragraphs which contain the term "hard drive" and seldom in general language, one may have a good guess at claiming it to be associated with computers.

It is difficult, however, to make absolute assumptions about the correlation of meaning or syntax with morpheme distributions – not only do natural languages offer possibilities to almost arbitrarily recombine morphemes and expressions, but there is additionally a great diversity of different languages. On top of that, we tend to use homonyms, synonyms and also metaphors – whether deliberately or as common idioms.

As such, for an appropriate distributional representation of words, one has to look at great corpora of text so that the influence of exceptional contexts ultimately fades out. Note that the observation of single words is in fact only a simplification, since the border

between morphemes and combinations of them is very hard to draw [12, § 2.2, page 152]; so words are usually only an approximation.

A more formalized view on the distributional hypothesis can be found in the following subsection.

2.1.1 Co-Occurrence Matrices

This section provides a formal definition and some practical considerations for co-occurrence matrices, which are a way to utilize the distributional hypothesis in practice.

2.1.1 Definition. Given:

- a discrete vocabulary $V = \{v_1, \dots, v_{|V|}\}$
- a corpus C with ordered elements $c_i \in V, i \in \{1, \dots, |C|\}$
- a mapping $w : \{1, \dots, |C|\} \mapsto \mathcal{P}\{1, \dots, |C|\}$ that for an element c_i defines its window for co-occurrence detection; usually, this is modeled by a window with fixed size sliding over the corpus, e.g. so that $w(i) = \left\{\max\{i - L, 0\}, \dots, \hat{i}, \dots, \min\{i + L, |C|\}\right\}$ with $L \in \mathbb{N}$ being the selected window radius, $i \in \{1, \dots, |C|\}$ [1]
- the indicator function $\mathbf{1}$ where for sets with cardinality 1 we also write $\mathbf{1}(a = b) := \mathbf{1}_{\{a\}}(b)$

We then define the co-occurrence matrix $COM(C) \in \mathbb{R}^{|V|^2}$ as follows:

$$COM(C)_{a,b} = \sum_{i=1}^{|C|} \left(\mathbf{1}(v_a = c_i) \cdot \sum_{e=1}^{|C|} \mathbf{1}(v_b = c_e) \cdot \mathbf{1}_{w(i)}(e) \right) \tag{2.1}$$

In a more probabilistic manner, this equation could also be understood as:

$$COM_p(C)_{a,b} = P(a, b \mid C) = \sum_{\mathbf{w}=1}^{|C|} P_{\mathbf{w}}(a, b \mid C) \cdot P(\mathbf{w} \mid C) \tag{2.2}$$

with $P_{\mathbf{w}}$ denoting the probability that words v_a and v_b appear in the same window $\mathbf{w}$ (0 if v_a is not the center word) based on the evidence in the corpus and $P(\mathbf{w} \mid C)$ the probability for any sample in the corpus to be contained in this specific window.

The usual definition for the window mapping w, for example also used for neural-network-based word embeddings [20], is quite intuitive: with a context radius L defined, the elements $c_{i-L}, c_{i-L+1}, \dots, c_{i-1}, c_{i+1}, \dots, c_{i+L}$ are taken into account for the window i [17].

[1] The broader definition of windows here does also allow sentence- or paragraph-based co-occurrence construction, by varying the window size per word.

We now covered the structure and construction of co-occurrence matrices, but it is yet to be specified what they are used for. As presented in equation (2.2), each cell of the co-occurrence matrix can be interpreted as a measure of probability for two words occurring in the same contexts based on an analysis of the corpus. The goal is that all high-valued entries of the matrix indicate a strong relation between the corresponding two words: intuitively, "high" and "mountain" should have a high co-occurrence rate, "whale" and "mountain" rather not.

For that objective, the co-occurrence matrix defined above is not yet a solution, though. While it would be, assuming that all words of the vocabulary appear with similar frequencies in the text corpus, this assumption is generally unrealistic: common verbs like "to be" or prepositions can be found in practically any text, while certain technical terms are rare in comparison. For the co-occurrence matrix, that would lead to rare words only occurring together having a lesser value than not particularly correlated but very frequent words due to their small overall counts, i.e. probabilities.

The next subsection introduces a metric which should solve that exact problem.

2.1.2 Pointwise Mutual Information

2.1.2 Definition. Let the PMI matrix $PMI(C) \in (\mathbb{R} \cup \{-\infty\})^{|V|^2}$ be defined as

$$PMI(C)_{a,b} = \log \frac{P(a, b \mid C)}{P(a \mid C) \cdot P(b \mid C)} \tag{2.3}$$

The *positive pointwise mutual information* metric offers the advantage of being sparse, by cutting off negative values:

$$PPMI(C)_{a,b} = \max\{PMI(C)_{a,b},\ 0\} \tag{2.4}$$

The PMI metric thus observes the frequency of two words appearing in one context as well as their frequency independently of each other and calculates the ratio. Accordingly, if two words are completely independently distributed, i.e. $P(a, b \mid C) = P(a \mid C) \cdot P(b \mid C)$, the corresponding entry of the PMI matrix will be 0; if correlated strongly it will be greater than 0, and if they never appear in the same context, the matrix contains $-\infty$.

Since the overall frequency is set in relation to the distribution of each word alone, we also avoid the problem of COM mentioned in Section 2.1.1, that it is impossible for rare words to achieve high values: if they are sufficiently strongly correlated, their PMI value will be positive nonetheless.

For a more detailed explanation of PMI, refer to [17, § 3.3] which also explores a few further optimized metrics like SPPMI.

2.2 Word Embeddings

When speaking of word embeddings, today it is often associated with neural-network-based embeddings such as word2vec[20]. The actual principle, however, only defines the embedding's goal: to map words to a (often continuous) vector space which offers possibilities of comparing or analyzing those automatically, e.g. similarity or relations between words.

An additional objective is to reduce the vector space's dimensions as much as possible, since it should also be viable to build and save embeddings upon corpora with billions of words; these will usually not consist of billions of *different* words, but uncompressed quadratic co-occurrence matrices may, depending on the number format used, quickly end up using some gigabytes alone. Assuming 4 bytes memory demand per matrix cell, e.g. as an integer, and a vocabulary of 50,000 words one would already reach a total memory of 10GB.

One possibility to reduce dimensionality for a set of datapoints is Principal Component Analysis (PCA) [24][10], or Latent Semantic Analysis (LSA) [27] specifically in the context of Natural Language Processing, both transforming the co-occurrence space by determining new dimensions best explaining variation in the data.

2.2.1 word2vec

In 2013, Tomas Mikolov et al. [20] introduced an efficient way to calculate word embeddings by novel neural network architectures which gained much popularity not only for their architecture, but also for the whole area of research.

For word2vec, neural networks are used to predict either the current window's center word (continuous bag-of-words model, CBOW) or the current center word's context (continuous skip-gram model). The training is based on the whole corpus to optimize the prediction accuracy, the network weights are eventually used as the resulting embedded vector. [20]

It showed that the vectors were not only applicable to standard similarity tasks, but also contained relational information; to repeat an example from the original paper, after training the model with a high number of dimensions on a large text corpus, when subtracting the vector of "Paris" from "France" and adding "Berlin" instead, the closest calculated vector turned out to belong to "Germany".

Accordingly, word2vec is quite powerful and, due to its high efficiency, has since its development been used for a diversity of applications, e.g. document ranking for search engines [23]. However, by relying on neural networks in an unsupervised manner, it is also susceptible to some risks, such as biases implicitly contained in the text corpus which can be very difficult to find and eliminate [4].

2.3 Explainability, Interpretability and Transparency

There is no fixed notion of what explainability and interpretability actually mean, so this thesis will base its definition on [7], where an overview about the most frequently used terms and their common ground or differences has already been elaborated.

While explainability and interpretability are sometimes used as synonyms, they will here be defined as two different concepts: interpretability is the possibility to completely map the respective model to a representation intuitively understandable to humans (e.g. graphs, diagrams or decision tables), whereas explainability describes the ability to present justification for any (but each by itself) specific decision of the model which has to be comprehensible to humans as well. [7, § II]

Additionally, we use the following notion of *transparency*:

> "Transparency is used as a synonym for model interpretability, that is some sense of understanding the working logic of the model." [7, § II]

2.4 Decision Trees

Since decision trees are a crucial part of the presented embedding approach, this section will present a short introduction on what they aim for, how they work and what risks they bear. Since the implementation created as a proof-of-concept for this thesis makes use of the Scikit-learn python package [25], the definitions and explanations in the following build upon the Scikit-learn documentation [1]; which in turn is based on the CART model [5].

2.4.1 Definitions (after [1])

2.4.1 Definition. Let $X \subseteq \mathbb{R}^n$ be the origin space where the training and test data come from and $Y \subseteq \mathbb{R}^l$ the label space, which the input data should be mapped to. For the scope of this thesis, l is set to 1 and for classification Y is usually defined as $\{-1, 1\}$, i.e. part of the respective class or "no evidence"; -1 instead of 0 mainly to be consistent with regression.

A decision tree effectively partitions the origin space until every terminal partition is maximally *pure*, i.e. there is no further partition allowed by the tree's parameters that could separate training data mapped to different labels any better.

The partitioning is repeated recursively, hence the name: starting from the *root node* subordinate nodes are added, each being assigned a part of the parent's classification space which again is separated for new child nodes. In the case of CART trees, the tree is

binary – every node chooses a feature and a corresponding threshold, which partitions its training samples as best as possible, so that each partition mostly contains samples with the same label.

2.4.2 Definition. Let $D_t = \{x_i \in X | i = 1, \dots, I_t\}$ be the set of training data and $L_t = \{y_i \in Y | i = 1, \dots, I_t\}$ the corresponding labels. The set of indices to samples contained in the classification space of node m is denoted as $I_t(m)$.

Then we call $\theta = (j, t), 1 \leq j \leq n, t \in \mathbb{R}$ a **split** which separates $I_t(m)$ into

$$I_t^{left}(m, \theta) = \{i \in I_t(m) \mid (x_i)_j \leq t\}$$

and

$$I_t^{right}(m, \theta) = I_t(m) \backslash I_t^{left}(m, \theta)$$

The objective for every node of the tree is to find the split adding the most information, that is, partitioning the observed training data in a way that the current node's left and right branches are as pure as possible. There are several criteria which can be used to evaluate a candidate split's quality, two of which are being defined in the following. Note that the given criteria are limited to classification trees which are most important to understand for this thesis. Common loss functions for regression trees, predicting not a discrete class but a continuous target variable, can be found in [1].

2.4.3 Definition. When observing our two labels $\{-1, 1\}$, the proportion of observations for each label c in node m is defined as

$$p_c(I) = \frac{1}{|I|} \cdot \sum_{i \in I} \mathbf{1}_{\{c\}}(y_i)$$

where $\mathbf{1}$ denotes the indicator function.

We furthermore refer to

$$H_G(I) = \sum_{c \in \{-1,1\}} p_c(I) \cdot (1 - p_c(I))$$

as the **Gini impurity** and

$$H_E(I) = - \sum_{c \in \{-1,1\}} p_c(I) \log\left(p_c(I)\right)$$

as the **entropy** or **information gain** metric.

For the selected measure H, a split's quality is then assessed by the impurity metric G with

$$G(m, \theta) = \frac{|I_t^{left}(m, \theta)|}{|I_t(m)|} H\left(I_t^{left}(m, \theta)\right) + \frac{|I_t^{right}(m, \theta)|}{|I_t(m)|} H\left(I_t^{right}(m, \theta)\right)$$

which is to be minimized.

After all these definitions, a quick sketch of the construction algorithm:

```
1  def build_tree(nodes, m, min_leaf):
2          θ* = arg min_θ G(m, θ)
3          I_left = I_t^left(m, θ*)
4          I_right = I_t^right(m, θ*)
5          if min{|I_left|, |I_right|} < min_leaf:
6                  return
7          nodes[m] = θ*
8          m_l = add_node(nodes, m, I_left)
9          m_r = add_node(nodes, m, I_right)
10         build_tree(nodes, m_l, min_leaf)
11         build_tree(nodes, m_r, min_leaf)
```

Algorithm 2.1: Top-down construction of a decision tree

2.4.2 Advantages and Risks

The main idea behind this thesis, choosing decision trees for learning word classes, is their generally high transparency, explainability and interpretability: the principles of decision tree construction are relatively intuitive, their prediction results simple to retrace and it is possible to draw or export graphs showing the tree structure in a comprehensible manner. In contrast to classical embeddings, one could see exactly which context words the decision tree takes as indicative for a specific class. For neural based approaches, this would be possible at maximum via expensive analyses – compare to [4] for an example to find and eliminate biases for only the gender subspace.

However, this does not come without drawbacks. For being able to retain the interpretability, a tree must not be too large, since otherwise it is hard to read and understand as well. Due to the model's simplicity, this is a quite drastic requirement which might not suit every dataset and objective.

This very problem is reinforced by the fact, that in practice usually heuristic algorithms are applied for decision tree construction. To cite the Scikit-learn documentation of the `DecisionTreeClassifier`'s [2] `random_state` parameter:

> "The features are always randomly permuted at each split, even if `splitter` is set to `"best"`. When `max_features < n_features`, the algorithm will select `max_features` at random at each split before finding the best split

[2] `https://scikit-learn.org/stable/modules/generated/sklearn.tree.DecisionTreeClassifier.html#sklearn.tree.DecisionTreeClassifier`

> among them. But the best found split may vary across different runs, even if `max_features=n_features`. That is the case if the improvement of the criterion is identical for several splits and one split has to be selected at random." [2]

Another problem is the overfitting tendency of decision trees – on especially large feature sets versus a small number of training samples, which we deal with here, the tree might end up classifying single examples by their unique feature combinations which are not at all generalizable. Therefore, we have another reason to keep the tree's depth low and choose rather high numbers for its `min_leaf_size` parameter. The CART model furthermore includes a pruning step which deletes nodes or entire parts of the tree after construction if they seem too specific.

Despite all of these countermeasures, overfitting as well as underfitting remain a huge problem and how successfully the Com2Sem approach deals with it will be reflected in Chapter 5.

2.5 Target Spaces

In this thesis, the term "target space" generally refers to a specific kind of embedding space (i.e., the vector space that co-occurrence vectors are mapped to) used for the proposed technique.

Developing an appropriate target space is even a crucial part of the approach, since mainly its structure is responsible for how suitable the respective Com2Sem model can be to a specific problem. Hereby, the target space does not only establish the number and meaning of features for resulting embeddings, but also a hierarchy in which they are included. The basic idea is to not grow a single multi-label classifier tree, but use a hierarchy of dual-class, low-depth decision trees instead. For instance, the feature Geographical Entity could possess several subordinate features such as City or Landmark which are only tested if their parent feature is already selected.

Also, rather than taking just the most probable result, the model may trace and output different routes in the hierarchy providing enough evidence (i.e. a prediction probability higher than a specified threshold). For example, observe the word "bat": it may appear with the words "live" or "species", given those are indicators of the Animal decision tree, but also "baseball" or "hit" which could be indicators e.g. for a Sports-Related and Weapon feature, respectively. Instead of selecting the route with maximum probability, one could return the resulting vectors of all three paths. Therefore, one has a way of handling homonymy which is also difficult in the context of classical embeddings.

2.5.1 Definition. A **target space** T is described by a tuple $(V, \mathcal{S})$ with a vector space $V \subset \mathbb{R}^{N_f}$, $N_f \in \mathbb{N}$ being the number of features, and the target space structure $\mathcal{S}$ which defines the hierarchy and meaning of features.

Hereby, let $\mathcal{S} = (p, s, d)$, where $p : \{1, \dots, N_f\} \mapsto \{0, \dots, N_f\}$ maps each feature to its parent feature with 0 denoting no parent, s is a bijection standing rather symbolically for the intended meaning attached to each feature (that is hardly possible to formulate mathematically and difficult even in natural language, so the semantics will be given via a descriptive name in future) and $d : \{1, \dots, N_f\} \mapsto \{\{-1, 1\}, \{-1\} \cup [0, 1]\}$ declares the domain for every feature. With $D_b = \{i \in \{1, \dots, N_f\} \mid d(i) = \{-1, 1\}\}$ we refer to the set of all binary/classifier features, and with $D_c = \{1, \dots, N_f\} \backslash D_b$ the set of all continuous/regression features.

Additionally, for every parent feature i, $i \in D_b$ must hold (since it is a yes or no classification).

2.5.2 Definition. An **embedding** in a target space T is a vector $e \in V$ satisfying following conditions $\forall i = 1, \dots, N_f$:

1. $e_i \in d(i)$
2. $(p(i) \neq 0 \wedge e_i \neq -1) \Rightarrow e_{p(i)} \neq -1$, i.e. a feature may only be selected if its parent feature is as well (as long as it has a parent)
3. $\exists j \in D_b (p(i) = p(j) \wedge i \in D_b \wedge e_j \neq -1) \;\Rightarrow\; e_i = -1$ [3], i.e. it is not allowed for two sibling features to both be selected

If a target space is defined accordingly, it can also allow for vector operations to solve certain word relation tasks. As a trivial example, given we have a very small target space with two features Instrument (1) and Size (2), a regression feature with 1 indicating "large", 0 "very small". Having defined the embeddings $\overrightarrow{\text{medium}} = (-1, 0.5)$, $\overrightarrow{\text{large}} = (-1, 1)$ and $\overrightarrow{\text{organ}} = (1, 1)$, one could calculate $\overrightarrow{\text{organ}} - \overrightarrow{\text{large}} + \overrightarrow{\text{medium}} = (1, 0.5)$ which would certainly be the vector that "piano" is mapped to. Still, this would probably also hold for a "drum kit" much less related to "organ" – to solve arbitrary word relation tasks, it would therefore be necessary to also add arbitrarily many specialized features.

2.5.3 Definition. To measure the similarity of two different embeddings without losing the target space's structural information, we want to define a new score expressing the relatedness of two embeddings.

[3] This is no general requirement, but postulated for reasons of simplicity since every vector does then possess at most one deep-most concept. As long as there are no conflicts between regression features, which would not be possible for a decision tree to predict from the same co-occurrence vector anyway, technically no separation into different vectors is necessary.

- D_b and D_c are defined as in Definition 2.5.1, i.e. the mutually exclusive sets of binary and continuous features.
- Let $C(E) = \{i \in D_b \mid \forall e \in E : e_i = 1\}$ be the set of common binary features of all embeddings contained in a non-empty, finite set E, where we also write $C(e_1, e_2, \dots)$ for short when $E = \{e_1, e_2, \dots\}$. Because of the second and third postulation in Definition 2.5.2, the features returned in $C(E)$ appear always in a coherent path.
- Analogously for regression features, we define the set of *comparable* features as $R(E) = \{i \in D_c \mid \forall e \in E : e_i \geq 0\}$ and apply the same short notation as for C. We call features comparable if they are non-binary and not -1 for any of the given vectors, since -1 effectively means *unknown* and we cannot compare with an unknown value.
- In the following, we denote the cardinality of a finite set with the $\#\cdot$ operator.

Then let

$$d_c(e_1, e_2) := \frac{\sum_{i \in R(e_1,e_2)} |(e_1)_i - (e_2)_i|}{\max\{\#R(e_1, e_2), 1\}} \tag{2.5}$$

be the (normalized) similarity score for only continuous features and

$$\mathrm{nco}(e_1, e_2) := \max\left\{ \frac{\#C\,(e_1, e_2) \;-\; d_c(e_1, e_2)}{\max\{\#C(e_1), \#C(e_2), 1\}},\; 0 \right\} \tag{2.6}$$

the *normalized concept overlapping* score. In case e_1 and e_2 share no comparable features, i.e. $\#R(e_1, e_2) = 0$, $d_c(e_1, e_2)$'s nominator is 0 and so is the subtrahend in $\mathrm{nco}(e_1, e_2)$.

After having defined the score function, some intuition for it should be provided. The initial idea is similar to the Jaccard index: with $\frac{\#C(e_1,e_2)}{\max\{\#C(e_1),\#C(e_2),1\}}$ we observe the number of shared classification features in relation to the maximum information being contained in any of both embeddings; regression features are not taken into account here, since they should rather denote additional information and are in general not mutually exclusive. To prevent the denominator from being 0 if none of the given embeddings have been assigned a concept, 1 is included in the max expression; $\#C\,(e_1, e_2)$ is then zero anyway.

Regression features should also contribute to the final result, but to a much lesser extent than binary key features; in particular, the more specific the individual vectors are, the more important we view the overall shared concept. Because of that, the division by $\max\{\#C(e_1), \#C(e_2), 1\}$ is applied to the subtrahend as well. Additionally, we divide by the number of comparable features to have more control over the term's contribution; regression features should only pose as a negative modifier, so they are normalized to a value between 0 and 1, including both end points. Again, we take the maximum with 1 in the denominator to prevent division by zero.

The original contribution of the regression features is determined by looking at the sum of differences in every comparable feature, i.e. $\sum_{i \in R(e_1,e_2)} |(e_1)_i - (e_2)_i|$.

2.5.4 Proposition. *Let e_1, e_2 be embeddings in T. Then:*

1. $nco(e_1, e_2) \in [0, 1]$
2. $nco(e_1, e_2) = nco(e_2, e_1)$, *i.e. normalized concept overlapping is symmetric*
3. *assuming e_1 and e_2 define at least one binary feature each,*

$$nco(e_1, e_2) = 1 \Rightarrow \forall i \in \{1, \ldots, N_f\} : (e_1)_i = (e_2)_i \ \vee \ i \in D_c \backslash R(e_1, e_2)$$

i.e. two embeddings score maximum similarity only if they differ at maximum in incomparable regression features and both possess at least one concept. Since vectors without any concepts do not contain useful information to base similarity upon, they score 0 similarity in any combination.

Proof. Let e_1, e_2 be embeddings in T.

(1) $\mathrm{nco}(e_1, e_2) \in [0, 1]$

$$\begin{aligned} C(e_1) \cap C(e_2) &= \{i \in D_b |\ (e_1)_i = 1\} \ \cap \ \{i \in D_b \mid (e_2)_i = 1\} \\ &= \{i \in D_b \mid (e_1)_i = (e_2)_i = 1\} \\ &= \{i \in D_b \mid \forall e \in \{e_1, e_2\} : e_i = 1\} = C(e_1, e_2) \end{aligned} \tag{2.7}$$

Thus, since $C(e_1) \cap C(e_2)$ is a subset of each $C(e_1)$ and $C(e_2)$ and accordingly $\#C(e_1, e_2) \leq \max\{\#C(e_1), \#C(e_2)\}$:

$$0 \leq \frac{\#C(e_1, e_2)}{\max\{\#C(e_1), \#C(e_2), 1\}} \leq 1$$

The subtrahend is always positive because the absolute values are used, so $\mathrm{nco}(e_1, e_2) \leq 1$.

The lower bound is trivially given by the outer maximum function.

(2) $\mathrm{nco}(e_1, e_2) = \mathrm{nco}(e_2, e_1)$

Considering C and R, e_1 and e_2 appear only in sets, just as $C(e_1)$ and $C(e_2)$ in the inner max expression, so the only difference could be caused by the nominator in d_c. However, since the absolute value distance is symmetric, $|(e_1)_i - (e_2)_i| = |(e_2)_i - (e_1)_i|$. Therefore, normalized concept overlapping is symmetric as well.

(3) Given e_1 and e_2 define at least one binary feature each,

$$\mathrm{nco}(e_1, e_2) = 1 \Rightarrow \forall i \in \{1, \ldots, N_f\} : (e_1)_i = (e_2)_i \ \vee \ i \in D_c \backslash R(e_1, e_2)$$

Let $\min\{\#C(e_1), \#C(e_2)\} > 0$ and $\mathrm{nco}(e_1, e_2) = 1$, then obviously $d_c(e_1, e_2) = 0 \Rightarrow \sum_{i \in R(e_1,e_2)} |(e_1)_i - (e_2)_i| = 0$, because it serves only as a negative modifier and the first

term cannot exceed the value 1 as shown above. That is: $\forall i \in R(e_1, e_2) : (e_1)_i = (e_2)_i$ with R being the set of comparable features just as stated.

Because the subtrahend is 0, $\#C(e_1, e_2) = \max\{\#C(e_1), \#C(e_2)\}$ is required, which is the case only if $C(e_1, e_2) = C(e_1) = C(e_2)$, as $C(e_1, e_2)$ is a subset of each $C(e_1)$ and $C(e_2)$. Accordingly, following the definition of C and condition (2) in Definition 2.5.2, $\forall i \in D_b : \ (e_1)_i = (e_2)_i$. Therefore, the stated implication holds.

If one of the word embeddings does not define any concept, $\#C(e_1, e_2) = 0$ and thus also $nco(e_1, e_2)$. □

Chapter 3

Trees, Targets, Transformation

3.1 Related Work

The idea of mapping semantic spaces to each other is not new; two approaches closely related to the one described here would be [8] and [14] which will be summarized shortly in the following. They in turn are related to [21] where a linear mapping is trained to map between distributional spaces of different languages, however, that fits the Com2Sem approach neither methodically nor with regard to the objective.

Both first mentioned papers perform a transformation of the distributional space into a well-defined target space; in [8], the space is a span of a set of feature-norms and therefore closest to the Com2Sem approach, while [14] maps distributional vectors to a model-theoretic space that should also be able to learn and define quantifiers. The target space in [8] is in general closest to the definitions in this paper, yet there are some important differences:

- The evaluation feature space is constructed upon the MacRae feature norms [19] and therefore high-dimensional, since this dataset includes a huge variety of features (as specific as `a_chicken` or `has_2_wheels` and as generic as `like_a_book` or `found_on_shelves`).
- The feature space is flat, i.e. defines no hierarchical structure. Every feature is predicted for itself.
- The distributional space is transformed linearly, with the coefficients learned using a training data set (again the McRae concepts, specifically).

Relying on the McRae norms led to some problems as stated in the aforementioned works:

> "However, overall the mapping looks too coarse. One reason could be the fact that the feature-based space is relatively sparse (the maximum number

> of features for a concept is 26, whereas there are over 2500 dimensions in the space). [...] It is important to note that a production frequency of zero for a concept-feature pair in the McRae dataset does not necessarily mean that the feature is not a plausible property of the concept, but only that it is not one of the most salient features, since it was not produced by any of the human participants (e.g.the feature `has_teeth` has not been listed as a property of CAT in the McRae dataset, but it is clearly a plausible property of the CAT concept). [...] This is yet another indication that the concept-feature pairs listed in the McRae dataset are not complete, meaning that there are salient features that apply to some concepts which have not been spelled out by the participants." [8]

These are some serious disadvantages, especially when training decision trees: following the CAT example, when the tree could notice that "teeth" does occur regularly with species possessing teeth, but also with CAT **not** specifying the respective feature, it could possibly lead to a disqualification of "teeth" as a discriminator and reinforce overfitting instead.

Therefore, other datasets will be used for the predefined target spaces; the Semantic Space (Section 3.3.2) training data was specifically designed for this purpose, while the SEMCAT dataset [28] is used for the automated tests based on the Topic Space (Section 3.3.3).

3.2 Com2Sem

3.2.1 Introduction

Inspired by the popular *word2vec* algorithm, the approach presented in this paper is named *Com2Sem*, *co-occurrence matrices to semantics*, since effectively a co-occurrence vector is mapped to a space directly encoding human understanding of semantics.

The goal is to develop a powerful yet transparent model for word embedding, providing complete insight in both construction and decision making; possibly even the search for contexts which contribute to the decision. Because of that explainability restriction, there are, however, some obstacles to overcome. Co-occurrence matrices grow quadratically with respect to vocabulary size, so that one may already reach memory limits based on a text corpus barely problematic for neural embeddings. There, one mainly needs to save the weight matrix of $|V| \cdot d$ where V is the vocabulary and d is the target dimension, usually several hundreds but still $d \ll |V|$ [20, § 4.3]. One common solution for this problem is to reduce the dimensionality, e.g. by feature selection or principle component analysis (PCA) [24][10]. To follow the explainability principle, however, the approach cannot actually rely on PCA which would make it more difficult to analyze the influence of specific context

words. Feature selection could be viable, but needs an automated or manual analysis beforehand, which context words can be ignored safely without affecting the training.

For that reason, an alternative method is used by constructing not one large co-occurrence matrix, but rather several small ones for separate parts of the corpus. In Section 3.2.2 the approach and its consequences are explained in detail.

As a transparent embedding method, the thesis relies on decision trees – which is partly problematic, since it grounds on the assumption that a relatively small set of single words in specific combinations is already enough to predict word classes. One part of the thesis is therefore to analyze and evaluate this exact hypothesis. The objective poses other problems for trees as well, e.g. the enormous amount of features (which could be relaxed by the fact that PPMI matrices tend to be relatively sparse [17, § 3.3]). For being able to keep the decision trees' depth low and accordingly their readability high while allowing for as many discriminating word contexts as possible (to partly neutralize the aforementioned drawback), the approach furthermore not only uses one, but several decision trees embedded in another tree-like structure: in fact, again some kind of decision tree in which nodes use the trained decision trees as a condition.

This section will cover the general process of how to apply the Com2Sem approach; from building the co-occurrence matrix and designing a target space to training and executing the model itself.

3.2.2 COM and PMI Construction

While the construction of a co-occurrence matrix might seem uncomplicated – just increasing a counter whenever two words appear in the same context – some important decisions have to be made beforehand.

Tokenizing

To begin with, we have to define what a *word* actually is. If we want to count every word form separately (e.g. "be", "is", "are", "been") that may lead to a very large vocabulary which in turn would increase memory usage drastically, especially in languages like German, Polish or Russian defining a variety of verb and noun forms. Additionally, we could miss information in cases where the form itself is not important at all – this point should vanish, however, when supplying enough training data for every variation.

Apart from that, observe the word "christmas tree". It clearly poses as one grammatical and semantical entity (if leaving out one of the words, the described concept would not be recognizable anymore), but according to the common notion of words as tokens separated by spaces or punctuation, it would be observed separately. On top of that, the notion of words can differ between languages: in the heavily context based Chinese languages they

could be very difficult to define or recognize, but the same applies to particle verbs as in German or Swedish, e.g. “angreifen” (German, “to attack”), which often appears in two parts in a sentence (“er greift an” ≅ “he attacks”).

For simplicity, composed words are ignored in this thesis. Another possibility, at least in languages related to English, would be to include a manually built or automatically obtained list of frequent composed words and bind them together – which is still no trivial solution. For example, one would also have to differentiate between “the book I took off the table” and “I just took off with the plane”.

Furthermore, one could use lemmatizers such as the WordNetLemmatizer (downloadable for the NLTK python package [18]) to unify different word forms for memory and performance purposes.

After having defined the kind of tokens contained in the vocabulary, the question arises, how a context should be defined. Usually, this is a fixed window (e.g. the four words before and after the current), but one could also use the whole sentence instead to allow for more topic-related than syntax-related entries. For the results showcased in Chapter 5, however, the sliding-window approach was used.

It is important to note, that all these decisions are no requirements for the Com2Sem model; this paragraph should rather inform about the choices that need to be considered. Com2Sem could in principle be used for any co-occurrence matrix.

Avoiding Large Co-Occurrence Matrices

As already noted in Section 3.2.1, the space complexity of conventionally built co-occurrence matrices lies in $\mathcal{O}(|V|^2)$, V being the observed vocabulary. This could be reduced by applying feature selection or dimensionality reduction (e.g. via a principle component analysis), however, that possibly also leads to the disregard of important features or missing explainability.

As an alternative, several small co-occurrence matrices for subsets of the corpus can be built containing less vocabulary. Accordingly, the space complexity does not grow quadratically anymore but linearly, since the effective size is the sum of each matrix size. It also offers the advantage that single matrices can be written to disk temporarily and do not have to remain in memory altogether. While speeding up the matrix computation and allowing more words to be observed for the same space limitations, the approach leads to higher computational costs when building the actual vectors.

But when (e.g. for local testing) it is more important to be able to build accurate vectors on a large corpus, that would exceed the RAM limits using a conventional COM, the latter is probably preferable.

It is also advisable to use some kind of matrix representation that does not require zero cells to take up space, since co-occurrence matrices are usually sparse, especially for rather small corpora (there are many words and tokens never occurring with each other). One very efficient solution for medium sized corpora based on the NLTK python package [18], which can also handle the calculation of PMI values as described in the next section, is presented in Appendix A.2.

Transforming Co-Occurrence Matrices

Using a plain co-occurrence matrix usually does not achieve the bests results since it allows fair comparisons only for center and context words which appear globally with roughly the same frequency.

Instead, transforming the matrix into a more meaningful format is advisable. A prominent choice is the PMI metric or its successors like PPMI or SPPMI; refer to Section 2.1.2 for more detailed information. It measures the number of center and context word co-occurrence in relation to their independent frequencies. An empirical approximation of the metric, as shown in [17], would be the following (COM as defined in Definition 2.1.1, $N_C = \sum_{u=1}^{|V|} \sum_{v=1}^{|V|} COM(C)_{u,v}$ the number of all observed co-occurrences):

$$PMI_{est}(C)_{u,v} = \log \frac{COM(C)_{u,v} \cdot N_C}{\sum_{v'=1}^{|V|} COM(C)_{u,v'} \cdot \sum_{u'=1}^{|V|} COM(C)_{u',v}}$$

If several COMs have been built, the calculation differs only slightly (for simplicity assuming that the vocabulary used for each co-occurrence matrix, determining its dimension, is the union of all individual vocabularies):

$$PMI_{est}\left(\bigcup_{k=1}^{K} C_k\right)_{u,v} = \log \frac{\sum_{k=1}^{K} COM(C_k)_{u,v} \cdot \sum_{k=1}^{K} N_{C_k}}{\sum_{k=1}^{K} \sum_{v'=1}^{|V|} COM(C_k)_{u,v'} \cdot \sum_{k=1}^{K} \sum_{u'=1}^{|V|} COM(C_k)_{u',v}} \quad (3.1)$$

To regain a sparse matrix, one could then apply the PPMI (positive PMI) format by cutting off all matrix entries lower than 0.

Note that the above formulas are based on the assumption that co-occurrence vectors should be constructed for **every** word in the vocabulary. Often, one may only be interested in e.g. nouns – in this case, co-occurrence and PMI matrices do not need to be quadratic.

3.2.3 Com2Sem Structure

The model's structure, i.e. the number and structural arrangement of its decision trees, depends directly on the defined target space (refer to Definition 2.4.1). For every feature (corresponding to a dimension) of the space there exists one decision tree determining

whether the respective feature applies to the given co-occurrence vector in the case of classifier trees, or a regression tree calculating the estimated value.

The selection of a classifier or regression tree is based on the given feature's domain $d(i)$.

3.2.4 Trees and Semantics

This section offers motivation for the approach and clarification on how exactly decision trees are supposed to recognize human perception of meaning.

In principle, the underlying hypothesis is a further restriction of the distributional hypothesis: we assume that not only a word's meaning is defined by all its contexts, but that some *parts* of its meaning could already be described by a rather small set of co-occurrences. Even more specific: this set of contexts holds not only for few words with the respective meaning but for most of them.

In other words: we want to find context words which, in specific frequencies of co-occurrence, determine a selected meaning. Some examples: words followed by "who" are mostly persons, after "a(n)" and "the" they are nouns or adjectives, appearing with adjectives such as "high", "thick" or "wide" indicates some physical shape.

A decision tree classifier now tries to find exactly those combinations of features which separate two classes in the data as best as possible (in our case: words with the supposed meaning and all others), hence it theoretically appears to be a good choice for our purpose. As presented in Section 2.4.2, there are however some difficulties with applying decision trees in this context; so by using a hierarchical model the overall accuracy should be improved.

3.2.5 Training

Training Algorithm

Training the hierarchical Com2Sem model works similarly to training a decision tree outlined in Section 2.4.1, particularly, it is applied recursively. As an overview:

```
1 def train_model(S = (p, s, d), s ∈ {1, ..., N_f}, D_t, L_t, I_t, trees):
2         if not ∃i ∈ {1, ..., N_f}: p(i) = s:
3                 return
4         for all i = 1...N_f: p(i) = s:
5                 trees[i].train({(x_j, (y_j)_i) ∈ D_t × d(s) | j ∈ I_t})
6                 train_model(S, i, D_t, L_t, {j ∈ I_t | (y_j)_i ≠ −1}, trees)
```

Algorithm 3.1: Construction of a Com2Sem model

Thus, the training data is split in every parent node of the model structure, so that children of a feature only have to train on data where the label embeddings contain every parent feature. This is helpful for keeping the tree depths low since the specialized trees in deeper levels do not need to repeat all the conditions of their parents: for example, if we had a feature VEHICLE and a sub-ordinate feature IS NAVAL in our structure, "water" or "ocean" could be a good discriminator for the latter without need to construct other conditions to exclude water animals, assuming those are correctly rejected by the VEHICLE tree.

A step not shown in *Algorithm 3.1*, since it is rather advisable than necessary, is to add a few purely zero samples labeled as -1 whenever training a classifier tree. That should prevent the tree from mainly applying negative conditions instead of positive ones which can particularly happen when the training data is not sufficiently balanced, i.e. far less negative than positive examples are given. By adding some artificial data that has to "flow" down the decision tree until the leftmost leaf, the probability of this event can be decreased a little. It does not prevent it completely, however, since the tree may select a split that allows nearly every word and then repeat the same thing with only the zero vectors filtered out beforehand.

If something like this happens, one should try adding more training data to other categories on the same level – if that still does not solve the problem, the chosen category might not be suitable, i.e. there are no appropriate positive discriminators the tree could choose from. Therefore, the target space would have to be redesigned.

Designing Training Data

Because decision trees are a supervised learning method and we want words to be classified as certain semantic categories, training data is necessary for a Com2Sem model. When designing the training data, one should keep in mind a number of things:

- Every feature should have a number of positive and negative samples. The number should at minimum be as high as the tree's `min_leaf_size` parameter, so that it is not forced to use impure nodes.
- Balanced training data is advisable to prevent heavy-sided growth of the respective decision tree, i.e. when instead of picking splits positively discriminating target class 1 it would sequentially sort out negative examples. The goal should be that a sample is selected as -1 (unknown) rather than incorrectly predicted as the given class.
- When speaking of the stereotypes contained in word embeddings, usually referring to neural word embeddings, it is important to note that these stereotypes are brought into the corpus by humans themselves. It is therefore fundamental to keep target space and training data clean from biases as much as possible.

- In principle, one and the same word could be used as a sample for more than one embedding. However, in the contradicting trees one would then obtain at least one impure leaf that possibly leads to overfitting (since the `min_leaf_size` requirement is met more easily) or even always rejecting the respective word by sorting it into a leaf with mostly negative samples. Accordingly, the approach should be avoided if possible.

- It would be advisable to only use frequent words in the training data, since their co-occurrence vectors are probably most "complete", statistically speaking, and as such contribute the most information.

When encountering wrong or questionable splits of a decision tree in the process of testing, the training data is the first place for counter-measures: by adding examples incorrectly (not) selected in the split, the decision tree can be forced to look for alternatives.

3.2.6 Embedding Prediction

For a prediction, the Com2Sem model's structure is traversed top-down and the traversal stops at every tree that does not classify the input as 1 with sufficient certainty. Regression trees are always applied to their parent's output vector; however, in the implementation their features can be overridden statically in lower depths for better accuracy (e.g. when selecting the feature Is Fish, the Lives in Water feature in their common parent Is Animal may always be set to 1).

Every maximal path (that is, every path that stops at a level where no classifier tree applies anymore, or a leaf) is then returned as an embedding. By being able to return multiple embeddings, homonyms can be modeled as well – which is also the reason why positive discriminators are desirable for all trees. While positively discriminating splits evaluate only the features assumed as representative for a class and as such do not lead to rejection of vectors appearing in additionally a whole other context, negatively discriminating paths could play off different aspects of meaning against each other.

As an example, we assume the vector for "bank" contains some high values for context words related to money and some for context words related to sitting, parks and nature (possibly also churches etc.). When the Economical Institution feature uses "money" to find candidates and the Garden Furniture feature uses "garden" and "sit", both trees do not interfere with each other and can be selected independently, hence capturing both meanings of "bank" correctly. If, on the other hand, Economical Institution uses the fact that these do not usually occur with nature-related words as opposed to scientific or environmental institutions, it might reject "bank" for its not related meaning aspect.

```
1  def predict(S = (p, s, d), s ∈ {0, ..., N_f}, x, trees, e, results):
2          for all i ∈ {1, ..., N_f}: p(i) = s ∧ d(i) ≠ {−1, 1}:
3                  e_i = trees[i].predict(x)
4          is_maximal = True
5          for all i = 1...N_f: p(i) = s ∧ d(i) = {−1, 1}:
6                  e^new = copy(e)
7                  e_i^new = trees[i].predict(x)
8                  if e_i^new = 1:
9                          predict(S, i, x, trees, e^new, results)
10                         is_maximal = False
11         if is_maximal:
12                 add e to results
```

Algorithm 3.2: Predicting on a Com2Sem model. Important: this sketch does only show the basic principles and does not implement all features of the actual algorithm.

3.2.7 Explainability of Com2Sem

Explainability is the approach's main objective – the model's function should be transparent and understandable, its results interpretable (and thus verifiable) by humans, and it should offer justification for any calculated embedding. In this section it is discussed whether and how these goals are implemented.

Transparency and Interpretability

In this thesis, transparency means the avoidance of black-box models, i.e. the idea of how the model works can be comprehensible to humans. According to [7, Figure 1], this can generally be assumed for decision trees, which recursively apply conditions to filter out inappropriate classes.

Additionally, by using the actual context words as features rather than vectors with reduced dimensions, we ensure *interpretability* – when encountering "state" ≤ 3.5 as a split condition inside the CONSTITUTION topic tree, one may deem it as suitable, while words not related to it at all can be viewed as inappropriate.

Justifications

However, sometimes the model's decision might not seem very obvious even though they work well in tests. Figure 3.1 shows an example that occurred very early in the implementation and testing phase, the representation of an IS CITY feature's tree:

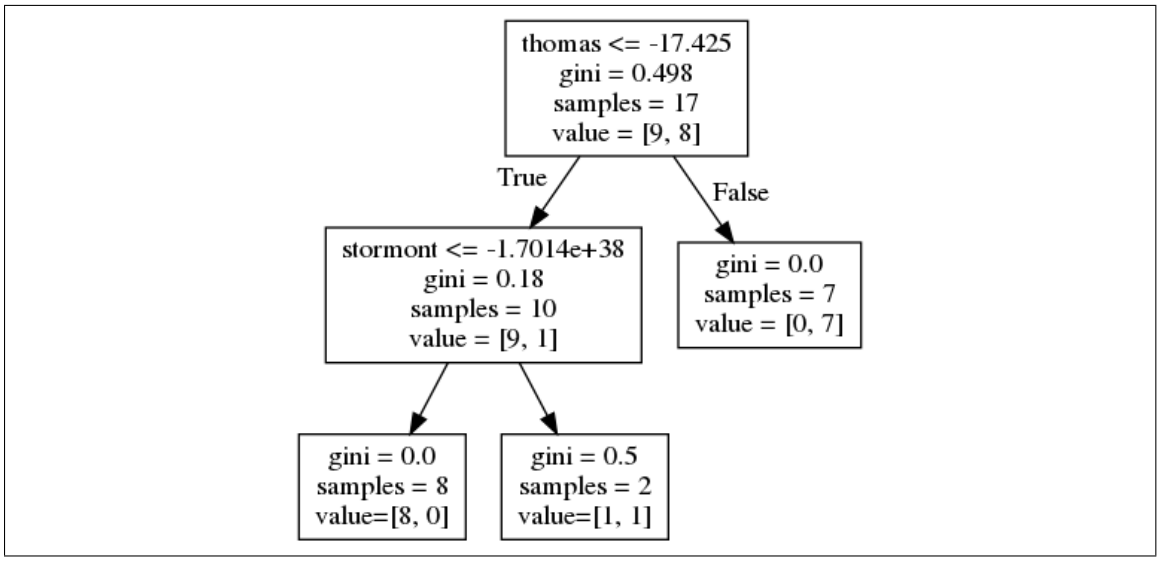

Figure 3.1: Clever Hans move on the Is City feature on an old version of the Com2Sem approach, still using PMI instead of PPMI.

We ignore the "stormont" split, since it is the result of overfitting (capturing only one positive sample whilst accidentally also recognizing a negative sample). Then, the main condition is that the word "thomas" occurs sometimes in the same context; note that even though with PMI a negative number denotes a negative correlation, i.e. they occur less frequently than they would if statistically independent, it is likely that for "Thomas" this holds in most cases.

Also, as is visible in the graph, 8 in 9 positive examples are correctly classified by the condition – further tests showed that it even correctly classified a high number of cities not specified in the training data. Since "Thomas" is not usually a name specifically connected with cities, some more research was necessary to explore why exactly the word is such a good discriminator.

As it seems, there are quite a lot of churches, hospitals and colleges named "St. Thomas", which on Wikipedia[1], the corpus on which the co-occurrence matrices were built, are usually listed right beneath their respective city.

While this is a valid discriminator, it does not suit the idea of a semantically motivated space – therefore it is recommended to use the PPMI metric which forces the tree to only look at positively correlated context words.

Two further points can be seen in this example:

- Due to the many aspects of meaning and context in every natural language, the tree structure and how it works may not always be clear to humans.

[1] `https://en.wikipedia.org/`

- However, since we know the target word, the context in question and also the underlying corpus, it is always possible to automatically search for shared contexts and show them to the user as a justification.

3.3 Designing Target Spaces

3.3.1 Applications

When talking about target space design, it is important to determine the area of application beforehand. Whereas the original co-occurrence vectors contain **all** the information we gained from the corpus and neural word embeddings at least many of them, just in a hard to analyze manner, the number and meaning of dimensions in the target space is completely user-defined in Com2Sem. Therefore, it is necessary to plan in advance the properties that word embeddings should possess and the way they are to be compared.

A few points to be considered:

- Since for a target space the information encoded is directly correlated with the number of dimensions and its structure, the more information the user needs to incorporate, the more complex the target space will grow. That in turn increases the effort needed to build a corpus of appropriate training data.
- It is only advisable to use the hierarchical structure, if it is sufficiently clear that there are indeed contexts discriminating **all** of the child features alike. If such properties can be assumed for the English language at all, is examined in this thesis by evaluating the highly structured Semantic Space presented in Section 3.3.2.
- It is not necessary to rely on *only* the Com2Sem embeddings – one possibility would be to use them combined with conventional word embeddings or co-occurrence vectors: instead of searching the whole vector space for similar vectors, the search could be limited to vectors contained in the same most specific category.

In each of the following two subsections, a target space is presented being designed for testing a specific field of applications.

3.3.2 Semantic Space

Application

The Semantic Space has been created mostly as a proof of concept; it should model concrete and abstract objects described by natural languages by selecting increasingly narrow categories. The idea is to examine, how well these theoretic concepts apply to actual natural language and can be reconstructed on basis of the distributional hypothesis.

Structure

The space's structure is based on [13, ch. 17.1], where language semantics are modeled as graphs, and concepts were assigned a so-called sort, arranged in a tree structure and defining "features" (not to be confused with the feature term of this thesis) – for example, the concept "rabbit" would be of sort "d" for "discrete" and possess the features ANIMAL+, HUMAN– and MOVABLE+ [13, fig. 17.2].

This thesis' adaptation combines both sorts and their features in the same hierarchy, sorts always as binary concepts and features as classifier or regression concepts depending on their meaning, and generally extends it afterwards to end up with more specific categories.

The training data was hand-designed for the Semantic Space, because of no known existing knowledge base that suited the space structure or could have been converted easily, and includes 395 entries at the time of writing. Note that not all of the features have been supplied training data if they were in general irrelevant for the evaluation performed in this thesis.

The lack of suitable existing datasets also complicates automatic testing of the model – benchmark approaches as proposed for the SEMCAT dataset [29] usually operate on a definition of similarity based on topics or word families, which partly overlaps with the Semantic Space ("bomb" and "mine" would in both views certainly belong to the same category), but not in all cases ("soldier" is also military-related in SEMCAT, while the Com2Sem model would ideally mark it as human profession and thus renders POTAG, potential agent, the closest common feature).

Nevertheless, three established intrinsic evaluators are applied to models learned on this space in Chapter 5 for being able to compare the results with those of common embedding approaches.

The full structure can be seen in Appendix A.1, here just two small excerpts:

- OBJECT: → Figure A.2
- SITUATION: → Figure A.13
- SITUATIONAL DESCRIPTOR: → Figure A.14
- QUALITY: → Figure A.15
- QUANTITY: → Figure A.16
- GRADUATOR: → Figure A.1

Figure 3.2: Top-level structure of the Semantic Space (only classifier features)

POTAG: potential agent of an action

- ANIMATE
  - MICROSCOPIC (bacterium, virus, ...)
  - ANIMAL
    - MAMMAL (dog, fox, whale, ...)
    - REPTILE (snake, lizard, ...)
    - BIRD (eagle, parrot, ...)
    - FISH (salmon, shark, ...)
    - AMPHIBIA (frog, salamander, ...)
    - INSECT (ant, bee, beetle, ...)
    - ARACHNID (spider, scorpion, ...)
  - PLANT
    - FRUIT (apple, pear, ...)
    - TREE (oak, willow, ...)
    - FLOWER (tulip, rose, ...)
    - SHRUB (bush, ivy, ...)
    - WEED (grass, moss, ...)
    - MUSHROOM (fungus, toadstool, ...)
  - HUMAN
    - PROFESSION (baker, priest, ...)
    - TITLE (king, lady, sir, ...)
    - STATE (sleepwalker, child, adult, ...)
    - GROUP (crowd, people, ...)
- INANIMATE
  - TRANSPORT (bus, ship, car)
  - CALCULATION (computer, calculator)
  - INDUSTRY (robot, machine)
  - HOME (mixer, dishwasher)
  - MILITARY (rocket, missile, drone)

Figure 3.3: POTAG subtree of the Semantic Space (only classifier features)

3.3.3 Topic Space

Application

The Topic Space was created for testing purposes on basis of the SEMCAT dataset[2] [29]. Rather than a precise representation of semantic concepts, it models only topical relations as described by the defined categories.

[2] `https://github.com/avaapm/SEMCATdataset2018`

In effect, a huge dataset can be used for evaluation on how well each tree recognizes new words for specific topics.

Structure

In SEMCAT, several thousand words are mapped to 110 different categories. The categories are rather broad, for example while the ANIMAL category contains mostly animal names, which by itself would be appropriate also for the Semantic Space, "nest" is contained there as well. Categories like BIRTHDAY are not strictly defined either (containing rather weakly related words like "age", "balloon", "born" and "day").

Every category, i.e. topic, is assigned one dimension containing exactly those words defined by SEMCAT. There is no hierarchy at all, neither regression features.

For the testing data then can be evaluated in how many of their correct (according to the dataset) categories they appear and how many false positives happen, allowing for an easily-interpretable measure of the model's prediction accuracy.

Chapter 4

Implementation

This chapter will on the one hand provide a short walk-through of the Com2Sem API, on the other hand in Section 4.4 it is discussed, how the model could be extended or its accuracy enhanced by integrating existing knowledge bases.

4.1 Target Space Definition

The target space must be defined using a YAML[1] file. The hierarchy can hereby be expressed by nested dictionaries, where the top-most components are those without parents. The `_uses` keyword is special, since it can be used to provide the regression features available for a specific parent feature as a list. As a very basic example:

```
1  _uses:
2      - movable
3  animal:
4      _uses:
5          - living on land
6      mammal: "-"
7      bird: "-"
8      fish: "-"
9  plant:
10     _uses:
11         - movable = 0
12     tree: "-"
```

Figure 4.1: Com2Sem config example

[1] https://yaml.org/

Hereby, the "–" just shows that no sub-features are available (in fact, every value not being a dictionary could be used here). The assignment in line 11 denotes that any embedding classified as PLANT will be marked as non-movable, overriding the output of MOVABLE's regression tree.

Additionally, not showing any effect in training or prediction, the "=∼" operator can be used instead of "=" to set a default value in the Training Data Creator (Section 4.2.2).

4.2 Defining Training Data

4.2.1 Format

The format for providing training data is a plain CSV file (comma-separated values), containing a column for the token and all feature columns; hereby, the feature columns are sorted alphabetically. The structure is thus contained only in the YAML file described in Section 4.1.

Rows starting with a "#" sign are ignored, as well as an optional header which for all but the first column must contain the respective feature names.

word	abstract object	abstr. temporal	action	always	amphibian	animal	...
fox	-1	-1	-1	-1	-1	1	...
car	-1	-1	-1	-1	-1	-1	...
table	-1	-1	-1	-1	-1	-1	...
height	1	-1	-1	-1	-1	-1	...

Figure 4.2: Example training data for the Semantic Space

4.2.2 Training Data Creator

Writing down the training data manually as a CSV takes quite some effort and is prone to errors for at least large target spaces.

To counter this problem, a simple graphical user interface relying on Tkinter[2] has been added for the purpose of creating and editing training data – the Training Data Creator.

[2] `https://docs.python.org/3/library/tkinter.html`

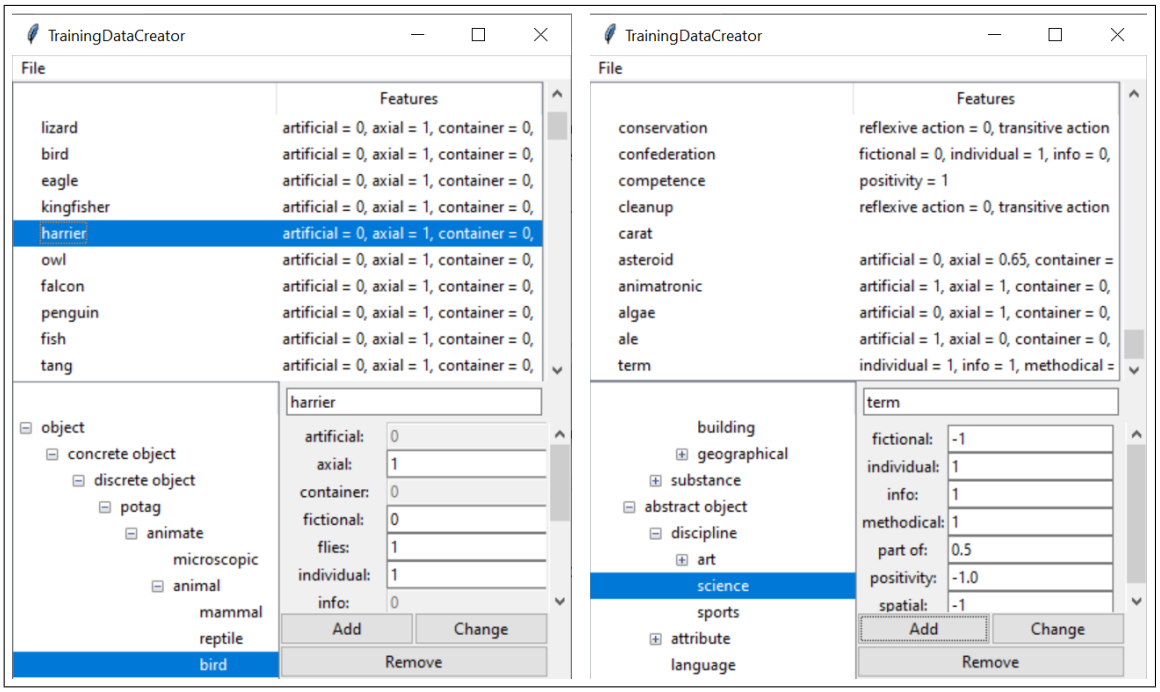

Figure 4.3: Screenshots of the Training Data Creator GUI

In Figure 4.3 some example screenshots of the tool are displayed: in the above frame one can see the list of word entries along with some of their defined properties, on the left the Com2Sem classifier structure is presented and on the right everything necessary to add a new or modify an existing entry.

The selected feature in the structure tree is used to set the classifier features, while on the right hand-side applicable regression features are shown and editable – the list of these is automatically updated when changing the selected classifier feature. Disabled entries denote that the respective feature is already explicitly set by the selected structure path; by default, every entry contains -1. In the structure's definition file, the $=\sim$ operator can be used to set another default without disabling the respective field.

Before using the Training Data Creator, a structure definition file must be selected; also, via the File menu or `Control-s` and `Control-o` training datasets can be saved or loaded.

4.3 Com2Sem API

4.3.1 Dependencies

The implementation is based on Python 3 [3] and additionally relies on some Python packages to implement its functionalities. These are most importantly NumPy [11] and Scikit-

[3] `https://www.python.org/`

learn [25], the latter of which provides the decision tree implementations of the model, as well as the PyYAML package[4] for parsing the structure configurations.

Furthermore, the NLTK package [18] and its provided wordnet corpus [22] are used for the co-occurrence matrix construction. That is, however, not an inherent dependency of the Com2Sem approach itself.

4.3.2 Com2Sem Class

All the main functionality is provided by one class, namely the `Com2Sem` class. For instancing, all feature names (i.e. words) of the original co-occurrence matrix and a path to the target space configuration file are required.

A model can then be trained using the `train(training_file, co_occurrence_lookup, co_occurrence_matrix)` method where `training_file` is the path to the training dataset (cmp. Section 4.2.1) as a string and `co_occurrence_matrix` is an arbitrary object (e.g. a `numpy.array`) which can be searched for words in the training data via the provided function `co_occurrence_lookup`.

Note that it is also possible to use an approach with multiple co-occurrence matrices here, as long as the lookup function is able to rebuild a single vector from it. The given mapping must furthermore return `False` if the word is not found in the matrix corpus, because the respective piece of training data is to be ignored in this case. The `co_occurrence_lookup` function should expect two parameters: at first the word to look up, then secondly the `co_occurrence_matrix` is just passed on.

For prediction, `predict(vector, threshold_function, regressor_wrapper, **kws)` is used which returns a list of different probable embeddings (by default using 0.5 as a threshold to select trees). If a customized threshold/decision function should be used, it can be given via the `threshold_function` parameter and expect a tree's probability as well as the tested feature's name. Additionally, all further keyword arguments are passed on to it. The optional `regressor_wrapper` may be given a function or lambda expression expecting the same arguments as a custom threshold function, just that instead of the tree's probability its predicted value is supplied. The function should return another numerical value replacing the one originally predicted.

By making use of the `sklearn.tree.plot_tree` function, a plot of any trained decision tree can be shown via the `show_tree(feature)` method. Furthermore, `justify` can be applied to a text corpus, a feature and a co-occurrence vector to retrieve a list of sentences which contribute to the classification result for the given feature.

[4] `https://pypi.org/project/PyYAML/`

4.4 Integrating Knowledge Bases

4.4.1 Potential Applications

Before discussing integrations and metrics, we have to recapitulate the actual applications, such as:

- similarity: do two words have a similar meaning?
- relational properties: given the relation between words A and B, which partner concept would model this relation most accurately for word C? Commonly, the word analogy relation is observed.
- automatic construction of interpretable word concepts: which properties would apply to a word's concept according to humans?

Similarity

Similarity tasks are one common application for word embeddings, where some metric is applied to the vectors and should optimally be a measure for how similar the respective words are – cosine similarity or simply the euclidean distance could be used here, for instance. The idea is directly implied by the distributional hypothesis, assuming that words appearing in similar contexts (i.e. possessing similar co-occurrence vectors) must be semantically related.

On the one hand, the Com2Sem approach allows for easier comparison of two vectors for humans and some kind of categorization, depending on the selected target space. On the other hand: unless using a large amount of dimensions and a very sophisticated structure, one will never reach the level of detail contained in conventional embeddings.

There are several possibilities of handling this disadvantage: One could use a hybrid similarity score function, combining both target and co-occurrence space information. Another way would be to filter words by target space categories and then order results by co-occurrence vector distances.

When relying on co-occurrence vectors once again, we however face the same problems as for classical Word Embeddings, such as biases and stereotypes; furthermore, we always have to store another (even though possibly reduced) vector representation for each word.

As an alternative, existing knowledge bases, e.g. WordNet [22], could be used instead of co-occurrence vectors to estimate the similarity of words in more detail. WordNet, for instance, contains semantical relations in a network, which can be used to evaluate the relatedness.

Relations

In word similarity tasks, a distance metric is applied that describes the relation of two words as a scalar.

For word relations, we want to have a vector as opposed to a scalar – a common example would just be the connection vector. Ideally, when two pairs of words each have a similar relation vector, they also share a semantical connection from a human point of view.

Once again, we differentiate between two levels of details for the Com2Sem approach. In principle, a model can offer possibilities of providing such relation measures – if its target space is designed accordingly. An example for that was given in Section 2.4. Nevertheless, just as with similarities, we lose many hidden dimensions by using a mostly discrete target space.

We must therefore use either co-occurrence vectors, possibly containing biases we do not know of, or suitable knowledge from other sources.

Concepts

Concepts are the actual strength, or rather purpose, of the model proposed in this thesis. The categories of concepts are defined in the target space and directly predicted by the model itself without need of further calculation or knowledge base consultation.

Nonetheless, to enhance the accuracy or sort out wrong classifications, it is possible to integrate other predictors or knowledge bases.

4.4.2 Integration

There are generally three places where the integration of external knowledge is sensible:

- the target space similarity score
- the decision mechanism for every node, i.e. feature selection on not only the decision tree probability but also external information
- filters to sort out inappropriate predicted vectors

The target space similarity score can be defined by the user directly, for example by combining the *normalized concept overlapping* (refer to Definition 2.5.3) with similarity estimations from WordNet. To accomplish the second point, the `predict` method offers an additional argument `threshold_function` allowing to use a custom selection mechanism: the given lambda expression or function should accept the probability for a positive match of the respective tree as well as the tree's feature name as parameters. Via the `**kwargs` further fixed arguments, such as the predicted word as string, can be passed onto the

measure. The return value must be `True` or `False`, determining whether the given concept is selected or not. Similarly, the `regressor_wrapper` parameter allows to override or sanitize the values predicted by regression trees.

The last method is a matter of post-processing and as such not integrated into the model directly, but trivial for the user to implement.

Chapter 5

Evaluation

5.1 Introduction

After having presented this thesis' approach, it is crucial to get a grasp on how it actually performs in practice; a goal, which this chapter is dedicated to.

However, before testing is possible, objectives are needed on which the model can be evaluated: those observed in the thesis are therefore described in Section 5.2. The paragraph does also list a number of obstacles which have to be dealt with when using Com2Sem.

Section 5.3 contains the results of the actual evaluation based on the two target spaces defined in Section 3.3: the Semantic and the Topic Space.

Finally, the observed results are set into relation with other Word Embedding methods such as word2vec to determine how well the Com2Sem approach is applicable to common NLP tasks.

5.2 Problems

5.2.1 Objectives

For evaluation to be possible, it is first necessary to decide on what to evaluate. In [32], where the performance of some popular word embedding models is reported for a set of different tasks, word similarity, word analogy, concept categorization and outlier detection beneath others were observed. Word analogy tasks cannot be adapted trivially to the observed target spaces, as mentioned in Section 4.4.1, so they are left out of scope for the evaluation.

For being able to compare the results of Com2Sem on the Semantic Space with those of [32], we choose to execute the evaluation analogously where possible, using for example

the same datasets. For the Topic Space based on the SEMCAT dataset, we apply Concept Categorization as well to also measure the amount of false-positives.

Word Similarity

For word similarity tasks one calculates a similarity score between two words' vector representations; which in turn can be compared to human-made scores [32, IV.A]. A common measure in conventional embeddings is the cosine similarity

$$\cos(v_1, v_2) = \frac{v_1 \cdot v_2}{||v_1|| \cdot ||v_2||}$$

but that is not well-suited to a mostly discrete target space defined in this thesis, since it does not make use of its hierarchical nature. Instead, a similarity score is proposed, the *normalized concept overlapping*, observing the deep-most common category whilst taking ranged values into account by using their differences as a negative modifier. The full definition and a more detailed explanation can be found in Definition 2.5.3.

For words that are predicted several embeddings, we observe the combination scoring best.

Word Analogy

This overview is given rather for the sake of completeness and not examined practically for reasons stated below.

The objective of word analogy tasks is, given two related words w_1 and w_2 as well as a new word w_1', to find a word w_2' related to w_1' the same way w_1 and w_2 are related [32, IV.B]. As an example:

$$\text{Paris} : \text{France} :: \text{Berlin} : \underline{\textit{Germany}} \quad [1] \tag{5.1}$$

Hereby, the first three words would be given and the underlined word is the one that should be found.

However, this task is rather problematic for the Com2Sem model. To enable such a specific answer, every word would have to possess a unique vector which would render the target space hardly readable. Otherwise, several words would be mapped to the same or a similar concept (possibly differing in regressed values), so we would always retain a *set* of possible answers which may be not of much help.

Instead, one could possibly use the original co-occurrence vectors (or another type of embedding) while applying the concept constraints as a filter on the search space. In

[1] Phrased as "Paris is to France, what Berlin is to Germany".

the above example (5.1) we would thus assume that w_2' must be a similar concept as w_2 ("France"), e.g. define the COUNTRY concept, and then a classical word analogy algorithm is applied to the configuration where only those w_2' are taken into account which are mapped to the COUNTRY concept.

Since Com2Sem would hereby pose as nothing more than an auxiliary tool, such an approach has not been implemented and evaluated for this paper.

Concept Categorization

Within a Concept Categorization task, word vectors should be separated according to their semantic categories into different groups [32, IV.C]. While those experiments usually require a clustering performed on a set of vectors, this is not necessary for Com2Sem vectors since they already clearly define the concepts they are contained in.

Nonetheless, concept categorization poses some challenges even without the necessity to perform clustering. For one, words can usually be clustered with respect to more than one property as stated in [32, IV.C]. While Com2Sem is able to find more than one meaning, so it would suffice to take the best fit, if a model finds too many vectors per word, that could distort the evaluation results.

Ideally, we would automatically test the whole defined target space; that, however, is generally impractical and not conducted for the Semantic Space in this thesis. It would be necessary to build a complete list of human-made word-to-concept mappings for this reason – which is especially difficult since humans are often not aware of all the different meanings a word could have. When being asked to assert the word "tiger" its meanings, one would also have to think of the military ships and helicopters named "Tiger", bands, albums, songs, places, products, organizations, persons etc. for a complete overview. The Com2Sem model did indeed spot some of them and classified "tiger" not only as "animal" but also as a "military inanimate potential agent" which would apply e.g. to ships, tanks and helicopters, beneath some other less obvious concepts which, however, mainly made sense when looking at the justifications.

In contrast, we try to apply the above approach for the Topic Space by using the SEMCAT [29] categories as supposedly complete word-to-concept mapping; however, this results in a more topic and less semantic related target space, so it is no ideal solution either.

Outlier Detection

For outlier detection, a number of words is given that all but one match a specific pattern (e.g. in "banana", "cucumber", "potato", "apple" and "oak", the last one is probably the outlier). It should then be viable to find the outlier by looking at the embedding vectors, using a *compactness score* [32, IV.D]:

> "Intuitively, the compactness score of a word is the average of all pairwise semantic similarities of the words in cluster W." [32, IV.D]

The compactness score $c(w)$ for word w is now adapted from [32, (12)] as following, $W = w_1, w_2, \ldots, w_{n+1}$ being the observed word cluster with one outlier and $e(w)$ mapping a word to its embedding vector:

$$c(w) = \frac{1}{n(n-1)} \sum_{w_i \in W \setminus \{w\}} \sum_{w_j \in W \setminus \{w, w_i\}} nco\left(e(w_i), e(w_j)\right) \tag{5.2}$$

The word achieving the maximum compactness score is the outlier[2].

5.2.2 Questions

This section should grant an overview about the main questions that should be answered by evaluating the model performance.

First of all, for the model to be sensible, we state the hypothesis that a rather small number of word contexts is enough to recognize a specific human-understandable concept. In extreme cases that could mean that a high co-occurrence score with only one word is already enough to recognize a word's meaning. The hypothesis is quite relevant since we restrict the number of observable contexts by the maximum tree depth, which in turn is necessary to keep the decision trees readable and the training dataset's size low.

Secondly, we want to know whether the Semantic Space's structure is an appropriate approximation for natural language semantics. If one of the low-level concepts (i.e. those near the root categories) is not recognizable from word contexts, many words are either falsely rejected or wrongly assigned the respective concept, leading to a poor overall performance. If such a problem arises and one uses enough training data, it should already be visible in the decision trees by causing many impure leaves.

Thirdly, we want to examine whether the Com2Sem approach shows any advantage or at least similar results in standard tasks when compared to common word embeddings.

[2] Note that in [32] erroneously the opposite is stated. However, the score as averaged similarity over all but the current word should be maximal if it is calculated over the actual cluster alone; that is the case if the outlier is the current word.

5.3 Evaluation

5.3.1 Semantic Space

For all experiments, the co-occurrence matrix was built on only a part of the corpus used in [32], due to memory and performance limits in the local testing environment; specifically, 1 GB of the wiki2010 corpus[3], for window size 4 just 500 MB. [4]

Word Similarity

The Word Similarity task was evaluated on WS-353-SIM [3] being based on the WS-353 dataset [9]. The tests were run several times, varying the window size taken into account for each co-occurrence matrix as well as `min_leaf_size` (minimum number of training samples each node must contain), `max_depth` (maximum depth of both tree types) and `criterion` (the measure used to evaluate a candidate split's quality in classifier trees, refer to Definition 2.4.3). As a criterion for regression trees, the default of Scikit-learn (`"mse"`) has been used.

As a measure of performance, Spearman's rank correlation coefficient, calculated using the function `scipy.stats.spearmanr`[5] included in the SciPy Python library [30], has been employed; additionally, the maximum and average number of meanings the model predicted for each word are shown in the figures.

Now to the evaluation: Most tests share at least a weak correlation with the human annotated similarities, even though far lower than the state-of-the-art results (Section 5.4). It must be stated, however, that in this evaluation one does not only deal with a new model architecture, but two different conceptions of similarity: while "train" and "car" should be mapped to the same concept TRANSPORT and thus achieve maximum similarity, they are given a similarity of only 63.1% in [9]. That is even more apparent with "monk" and "slave" – both having at least near maximum similarity according to *normalized concept overlapping*, since they should possess HUMAN as common category, while being listed with a similarity of 9.2%. There are several of these discrepancies between dataset and Semantic Space structure, so the maximally possible correlation is already considerably limited.

For being able to train a model achieving very high correlation with humans, another target space would have to be created – whether one suiting the objective actually exists, cannot yet be answered.

[3] `http://nlp.stanford.edu/data/WestburyLab.wikicorp.201004.txt.bz2`

[4] On a side note, all datasets used for the Semantic Space evaluation had already been collected in a Github repository at `https://github.com/vecto-ai/word-benchmarks`.

[5] `https://docs.scipy.org/doc/scipy/reference/generated/scipy.stats.spearmanr.html`

Window	`min_leaf_size`	`max_depth`	`criterion`	Max./Avg. Meanings	Result
3	2 / 1	2	entropy	8 / 4.62	8.18
3	2 / 2	5	entropy	7 / 2.28	23.24
3	2 / 2	5	gini	6 / 2.19	16.78
3	2 / 2	10	entropy	8 / 2.25	22.52
3	2 / 2	10	gini	6 / 2.06	26.09
3	2 / 2	15	entropy	8 / 2.27	22.15
3	2 / 2	15	gini	7 / 2.15	25.13
5	2 / 2	2	entropy	10 / 4.44	24.31
5	2 / 2	2	gini	13 / 4.04	26.69
5	2 / 2	5	entropy	10 / 2.44	14.15
5	2 / 2	5	gini	9 / 2.64	21.68
5	2 / 2	10	entropy	9 / 2.29	20.89
5	2 / 2	10	gini	10 / 2.49	19.78
5	2 / 2	15	entropy	8 / 2.22	7.68
5	2 / 2	15	gini	10 / 2.65	15.69

Figure 5.1: Evaluation of Com2Sem on Word Similarity tasks; note that for this experiment, the window is aligned to the center, so that both preceding and following words are taken into account. The `min_samples_leaf` parameter is defined separately for (a) binary and (b) regression trees.

Window	min_leaf_size	max_depth	criterion	Max./Avg. Meanings	Result
2	2 / 1	2	entropy	8 / 4.10	5.79
2	2 / 1	5	entropy	6 / 1.98	33.25
2	2 / 2	5	entropy	6 / 1.98	33.17
2	2 / 2	5	gini	6 / 1.98	22.64
2	2 / 2	10	entropy	6 / 2.11	24.71
2	2 / 2	10	gini	8 / 2.15	25.35
2	2 / 2	15	entropy	6 / 2.10	26.36
2	2 / 2	15	gini	6 / 2.17	30.41
2	2 / 2	20	entropy	6 / 2.10	27.15
2	2 / 2	20	gini	6 / 2.17	31.68
2	2 / 2	30	entropy	6 / 2.10	26.95
2	2 / 2	30	gini	6 / 2.17	31.77
3	2 / 2	2	entropy	6 / 2.66	22.44
3	2 / 2	5	entropy	5 / 2.12	19.62
3	2 / 2	5	gini	7 / 1.90	29.32
3	2 / 2	10	entropy	6 / 1.94	19.59
3	2 / 2	10	gini	6 / 1.75	17.94
3	2 / 2	15	entropy	6 / 1.99	16.41
3	2 / 2	15	gini	6 / 1.75	18.01
4*	2 / 2	2	entropy	14 / 3.77	16.17
4*	2 / 2	5	entropy	7 / 2.24	6.55
4*	2 / 2	5	gini	24 / 2.34	7.72
4*	2 / 2	10	entropy	7 / 2.34	-3.26
4*	2 / 2	10	gini	21 / 2.38	4.03

Figure 5.2: Evaluation of Com2Sem on Word Similarity tasks; note that for this experiment, the window is left-aligned, so that only following words were taken into account. The `min_samples_leaf` parameter is defined separately for (a) binary and (b) regression trees.
(*) The embeddings were built on a 1 GB corpus taken from Wikipedia for window sizes 2 and 3; however, due to memory and performance limitations in the testing environment, the co-occurrence matrix for window size 4 has only been collected over 500 MB of the corpus.

While the first entries in Figure 5.1 and Figure 5.2 show that a tree depth of 2 is not enough to capture any similarity information, it appears that with a depth of 5 one already achieves quite good results; specifically, the correlation even decreases for depths between 5 and 15. Also notably, the split criterion's choice seems to affect the outcome remarkably – which is interesting, since as stated in [16]:

> "We found that they [Gini and Information Gain criterium] disagree only in 2% of all cases, which explains why most previously published empirical results concluded that it is not possible to decide which one of the two tests performs better." [16]

It might be that the large impact of a single different decision in classifier trees near the root on the overall similarity leads to this phenomenon; however, that seems unlikely as the observations in Figure 5.3 show similar behavior for co-occurrence vectors reduced to only those features selected in a specific Com2Sem model. Still, this question shall remain out of the scope for this thesis.

While the correlation results for low depths are comparatively high, looking at the common concepts they yield reveals that the concept mapping itself is not quite as accurate (e.g. "tiger" being classified as SUBSTANCE among others [6]). One possible explanation for this could be that the conditions found are not complex enough to capture the actual strict categories, instead the trees itself would classify only loosely related word classes, which are more suitable to the test datasets' notion of similarity. When the trees grow more complicated using higher depths, this property would probably diminish, which could explain the drop in correlation between 5 and 15, until the model starts to become better in performing its actual objective.

We also observe that a larger window size seems to influence the performance negatively: in both Figure 5.1 and Figure 5.2 the overall correlation decreases with increasing window size (especially related to high maximum depths), while we receive relatively good results with a maximum depth of 2 already. On the other hand, the model shows better results for concept categorization when trained on larger windows (refer to the next section and Section 5.3.2). Looking at the tables Appendix A.3 and Appendix A.4 suggests that the problem could be the space structure: with higher window sizes the trees tend less to underfitting, so that more words are rejected than accepted, and due to the low accuracy this also includes correct words. If more words are rejected in low levels of the target space, they include less similarity information, and as such correlate badly with the human annotations.

[6] While there exists "tiger balm" which could theoretically justify the result, this expression only occurred once in the observed corpus and is as such unlikely to change the prediction outcome that strong.

Method	Window	`min_leaf_size`	`max_depth`	`criterion`	Spearman	Pearson
cos*	2	–	–	–	46.46	45.05
cos	2	2 / 2	2	entropy	34.47	36.20
nco	2	2 / 2	2	entropy	5.68	5.77
cos	2	2 / 2	2	gini	33.03	34.27
nco	2	2 / 2	2	gini	10.68	10.90
cos	2	2 / 2	5	entropy	41.47	42.20
nco	2	2 / 2	5	entropy	33.74	36.27
cos	2	2 / 2	5	gini	38.73	39.39
nco	2	2 / 2	5	gini	23.06	23.44
cos	2	2 / 2	10	entropy	38.48	39.08
nco	2	2 / 2	10	entropy	26.54	27.20
cos	2	2 / 2	10	gini	36.82	37.72
nco	2	2 / 2	10	gini	25.52	27.29
cos	2	2 / 2	15	entropy	38.67	40.00
nco	2	2 / 2	15	entropy	28.73	29.83
cos	2	2 / 2	15	gini	36.84	38.10
nco	2	2 / 2	15	gini	33.17	33.98

Figure 5.3: Evaluation of Com2Sem on Word Similarity tasks; note that for this experiment, the window is left-aligned, so that only following words were taken into account. The `min_samples_leaf` parameter is defined separately for (a) binary and (b) regression trees. In this experiment, not only the normalized concept overlapping has been calculated but also the cosine similarity on the respective co-occurrence vectors, ignoring all features not being used in any split of the Com2Sem model.
(*) Correlation using the complete co-occurrence vectors.

For the sake of comparison, another experiment has been conducted, the results of which are displayed in Figure 5.3: instead of using the normalized concept overlapping, we apply cosine similarity to the co-occurrence vectors directly to measure their correlation with the annotations, however, observing only those features which occur at least in one split of the Com2Sem model. In the first row, we can see the correlation on the original co-occurrence vectors.

As it appears, the model results for reduced co-occurrence vectors as well as target space embeddings turn out better when observing the Pearson correlation coefficient[7] instead of the Spearman correlation coefficient: the best configuration scores 36.27% in this case, and is thus about 9 percentage points lower than the original co-occurrence vectors' result. Considering the different perceptions of similarity and that we transformed vectors with more than 1.5 million features into a mostly dual-valued form with only about 170 features, that is remarkable, though. Still, the results are not satisfactory for practical purposes.

The reduced co-occurrence vectors in contrast perform quite well: the best correlation (achieved by the model that also scores best using normalized concept overlapping) is almost as high as the original correlation while only about 450 features are contained. This introduces a new application to the Com2Sem approach: a transparent way to reduce the dimensionality of a distributional space. It could be subject to further research how the performance changes when varying the defined target space.

Concept Categorization

For evaluation purposes we use the large BM dataset [31], containing 5,231 words each mapped to one of 56 semantic categories. Ignoring words that could not be found in the corpus with sufficient frequency or were already contained in the training data, 3,452 words remain.

While performance in this task is usually measured by a cluster purity metric, we observe accuracy instead as clustering is not necessary in our case. Each one of the 56 categories is mapped to a corresponding feature of the Semantic Space; the mapping is not bijective because the Semantic Space features are rather general, so "girls first name" and "males first name" are both mapped to HUMAN. The complete mapping can be found in Appendix A.3.

In addition to the overall accuracy, we measure the maximum and average number of meanings predicted for a word, as the ability to capture more than one meaning could distort the results. In an extreme case, if the model predicted all concepts to every word, the accuracy would be 100% even though one gains no information at all.

[7] `https://docs.scipy.org/doc/scipy/reference/generated/scipy.stats.pearsonr.html`

Window	`min_leaf_size`	`max_depth`	`criterion`	Max./Avg. Meanings	Accur.
2	2 / 2	1	entropy	13 / 2.90	14.87
2	2 / 2	1	gini	11 / 2.16	10.40
2	2 / 2	5	entropy	9 / 1.31	4.57
2	2 / 2	5	gini	11 / 1.24	8.47
2	2 / 2	15	entropy	9 / 1.35	8.81
2	2 / 2	15	gini	9 / 1.26	8.47
3	2 / 2	1	entropy	8 / 2.31	20.90
3	2 / 2	1	gini	9 / 1.90	15.69
3	2 / 2	5	entropy	9 / 1.35	11.48
3	2 / 2	5	gini	8 / 1.40	13.48
3	2 / 2	15	entropy	8 / 1.36	12.56
3	2 / 2	15	gini	8 / 1.35	13.25

Figure 5.4: Evaluation of Com2Sem on Concept Categorization tasks; note that for this experiment, the window is left-aligned, so that only following words were taken into account. The `min_samples_leaf` parameter is defined separately for (a) binary and (b) regression trees.

As clearly visible in Figure 5.4, the overall performance is not particularly high; anyhow, it is more sensible to observe the accuracy per cluster directly which is therefore displayed in Appendix A.3. That is because in contrast to the clustering approach, we do not try to separate the given words into 56 different categories, but they are mapped to the 172 target space features where usually only the most specific ones are chosen as valid (e.g. "tomato" is only recognized correctly if it is contained in the FRUIT class, the PLANT feature does not suffice) [8].

Another problem lies within the choice of training data – for some categories of the dataset no or only few training data has been supplied originally. However, when looking at Appendix A.3, where the number of concordant training samples per category is shown as well, there are both categories with *no* training data supplied achieving high accuracy and categories with many training samples scoring rather low. Therefore, it appears as if the diversity of training samples for each feature is more important than many examples for a specific category.

[8] The INSTRUMENT feature is one of the exceptions here, which has been selected since the dataset sometimes contains examples from different categories, for example "table" (FURNITURE) as well as "sponge" (DEVICE) in the "kitchen utensil" cluster.

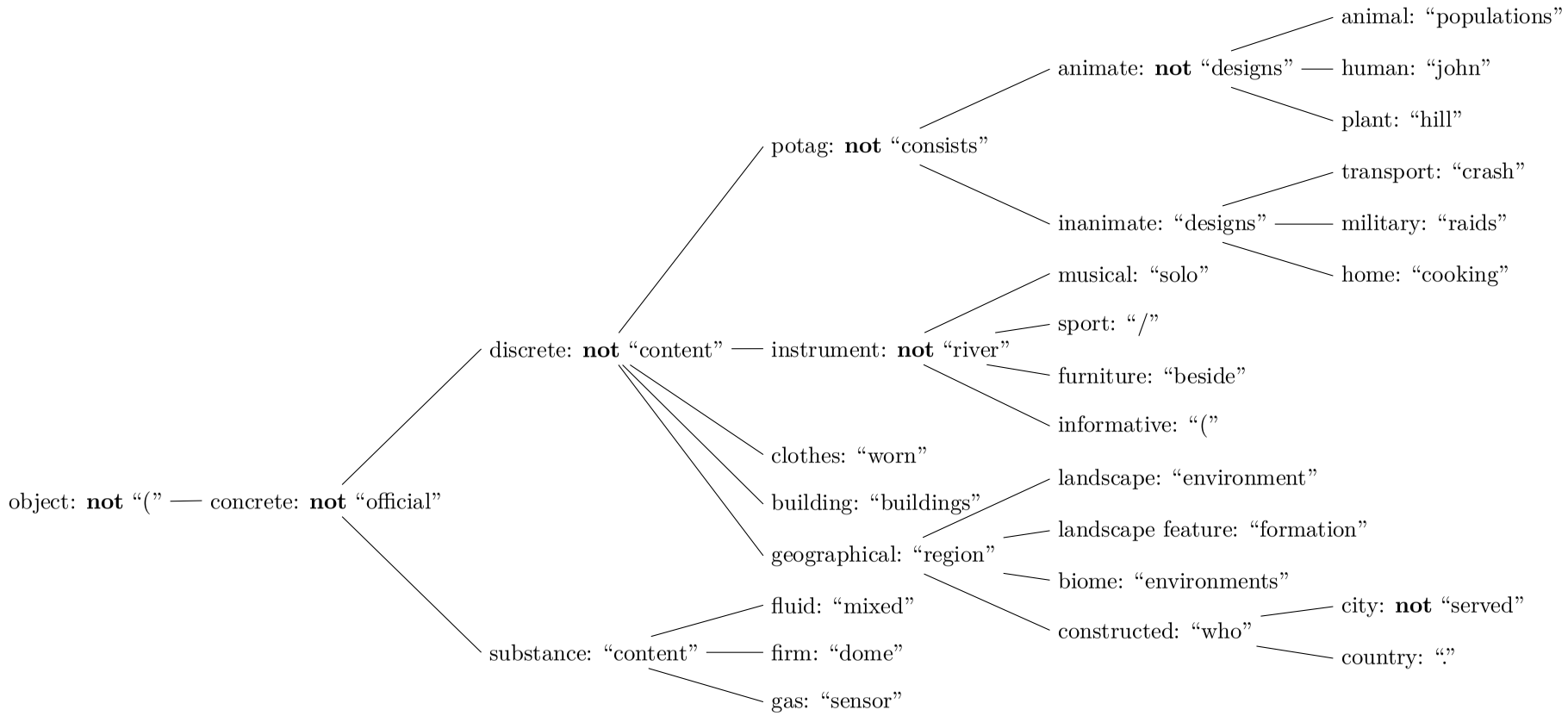

Figure 5.5: Excerpt from a Com2Sem model with depth 1 using the Information Gain criterion and `min_leaf_size=2`. The figure only shows which context words are observed as indicators for the membership to the respective concept; the individual thresholds are ignored.

Outlier Detection

In Figure 5.6, some results for different model configurations and window sizes are listed, showing the maximum number of detected meanings (i.e. vectors per word), the detection's accuracy and the Outlier Position Percentage (OPP) [26, eq. (5)]. The last one denotes the average normalized position over all outliers when cluster and outlier are sorted according to their achieved compactness score.

As a dataset for testing, WikiSem-500 [26] and 8-8-8 [6] should have been used, but there turned out to be some problems:

- Both datasets contain a large number of composed words, such as `population_density` or `Baden_Airparks`, not being collected for the model.
- Quite a number of words is not contained in the observed corpus (at least with sufficient frequency) and as such these do not allow for fair comparison.
- Just as with word similarity, the space structure is partly not sophisticated enough to correspond with the datasets; for example, in 8-8-8 one finds the "Information Technology companies" cluster with outliers such as "Boeing". The Semantic Space, however, does not differentiate between different kinds of companies. Even if they were predicted correctly, these companies should be contained in the ECONOMICAL INSTITUTION concept and define no clear outlier.

To confront the first two points, the respective words are simply ignored and so are clusters not containing at least two elements afterwards. While this allows for a better model-specific evaluation, it makes it difficult to compare with the results in [32]. In effect, we retain 66 lines (with mostly reduced outlier and/or word sets) of WikiSem-500 for models based on a window-size-2 co-occurrence matrix and 78 lines for window size 4.

The rather small 8-8-8 dataset is excluded completely from the evaluation.

Unexpectedly, the best results of all performed tests occur with a co-occurrence matrix taking only one context word into account, whilst allowing only one single split, as is visible in Figure 5.6.

Since we conduct the experiment on only about 10% of the original dataset with a reduced outlier or word list for most clusters, the accuracies we achieve are notably higher than in [32, Table V]. In contrast, the Outlier Position Percentage is far lower – which might seem counter-intuitive, however, if many clusters include not more than four words instead of the original five to seven, even one position off would already result in an OPP of 50 to 75 percent. Accordingly, comparing the results of this paper and [32] is not reasonable, but we do have weak evidence that the model is applicable to the Outlier Detection task – even with minimal complexity of the tree configuration.

Window	`min_leaf_size`	`max_depth`	`criterion`	Max. Meanings	Acc.	OPP
2	2 / 2	1	entropy	8	35.77	63.90
2	2 / 2	1	gini	8	35.77	63.90
2	2 / 2	2	entropy	10	35.04	58.37
2	2 / 2	2	gini	10	34.31	59.54
2	2 / 2	3	entropy	5	29.20	59.93
2	2 / 2	3	gini	7	29.93	59.47
2	2 / 2	5	entropy	5	29.20	59.93
2	2 / 2	5	gini	7	21.90	51.76
2	2 / 2	10	entropy	5	22.63	52.79
2	2 / 2	15	entropy	6	21.90	53.96
2	2 / 2	15	gini	7	29.20	59.89
2	2 / 2	30	entropy	5	21.90	54.11
2	2 / 2	30	gini	7	29.20	59.89
3	2 / 2	1	entropy	13	28.49	54.43
3	2 / 2	1	gini	14	20.93	52.36
3	2 / 2	2	entropy	10	18.60	54.32
3	2 / 2	2	gini	8	18.60	53.36
3	2 / 2	3	entropy	8	25.00	56.21
3	2 / 2	3	gini	9	24.42	57.93
3	2 / 2	5	entropy	7	19.19	52.19
3	2 / 2	5	gini	7	17.44	47.83
3	2 / 2	10	entropy	7	27.91	59.18
3	2 / 2	15	entropy	7	27.91	58.76
3	2 / 2	15	gini	7	20.35	51.92
3	2 / 2	30	entropy	7	27.91	58.76
3	2 / 2	30	gini	7	18.60	51.24

Figure 5.6: Evaluation of Com2Sem on Outlier Detection tasks; note that for this experiment, the window is left-aligned, so that only following words were taken into account. The `min_samples_leaf` parameter is defined separately for (a) binary and (b) regression trees.

5.3.2 Topic Space

For the topic space, we only evaluate concept categorization, because it is not complex enough for other intrinsic evaluators. There are two main differences to the Semantic Space experiment in Section 5.3.1:

- The target space does not focus on semantic relations but topics.
- As the mapping of SEMCAT categories and Topic Space features is bijective, we can also determine the number of words erroneously predicted for a given category. This is based on the assumption that SEMCAT is complete, i.e. each word is specified for every category it could possibly be contained in.

SEMCAT includes 6,559 different words which all occur in at least one of 110 categories [29]. For the evaluation, we observe only a part of those, specifically 48 categories still containing 5,167 words. The first twenty words of each category are selected as training samples.

The co-occurrence matrices are built on 100 MB of the wiki2010 corpus[9]; the window covers – depending on the experiment – 2, 3, 7 or 21 words, including the center word. For window size 2 the right-hand context word is used, in all other cases the window is centered. Additionally, for only window size 21 the corpus is cleaned beforehand to reduce memory usage: stop words[10] and punctuation are removed completely, additionally the `nltk.stem.WordNetLemmatizer`[11] is used to lemmatize all words and therefore reduce the vocabulary size.

The Com2Sem model's parameters are fixed to `min_samples_leaf=2` and `random_state=0` whereas `max_depth` and `criterion` remain changeable.

While the first results displayed in Figure 5.7 already seem promising at first glance, achieving an accuracy of almost 60%, they are an indicator for problems of the tree when looking at the number of wrong classifications (refer to Appendix A.4). Many of the trees are too generalized and thus classify more than half of all words incorrectly.

This tendency fades when the tree is allowed to use more than one split, but in trade-off with the overall accuracy. Interestingly, the criterion's choice still makes a difference, where entropy seems preferable for low depths and gini achieves slightly better results for high depths. We also notice that increasing the window size helps to improve the results (for `max_depth=1` it leads to less general trees).

[9] `http://nlp.stanford.edu/data/WestburyLab.wikicorp.201004.txt.bz2`

[10] Using `nltk.corpus.stopwords` for the English language (`stopwords.words('english')`). Refer to `https://www.nltk.org/book/ch02.html` for information about NLTK corpora.

[11] `https://www.nltk.org/api/nltk.stem.html#nltk.stem.wordnet.WordNetLemmatizer`

Window Size	`max_depth`	`criterion`	Accuracy
2	1	entropy	57.59
2	1	gini	48.50
2	15	entropy	10.07
2	15	gini	13.23
3	1	entropy	45.35
3	1	gini	42.27
3	15	entropy	11.51
3	15	gini	13.21
7	1	entropy	43.76
7	1	gini	40.75
7	15	entropy	12.67
7	15	gini	13.64
21*	1	entropy	55.69
21*	1	gini	56.06
21*	15	entropy	21.22
21*	15	gini	21.43

Figure 5.7: Evaluation of Com2Sem on topic-related categorization.
(*) In the last run, lemmatization and stop-word removal have been applied to the corpus before collecting the co-occurrence matrix.

Observing the accuracies and false-positive counts per category in Appendix A.4 shows that the prediction performance depends strongly on the respective category: while for the PIRATE concept a high accuracy is only achieved with at least about 400 false-positives, the COUNTRY concept reaches 58.75% accuracy with only 44 false-positives. The LANGUAGES category achieves quite good results as well (for small window sizes using "language" as a primary indicator word), with a window size of 21 even 91.58% accuracy while selecting a (comparatively) small amount of 180 words incorrectly.

As COUNTRY and LANGUAGES are the most consistent SEMCAT categories, i.e. related semantically rather than topically, it suggests that the Com2Sem features should be chosen carefully in dependence of the selected context. For topic based categorization, instead of using words out of a local window, the paragraph- or document-based context might be more helpful; as visible in Figure 5.7, the best results (i.e. high accuracy, less underfitting) have been achieved with a window size of 21.

5.4 Comparison and Summary

Eventually, the results of the performed tests are compared to those elaborated of Wang et al. [32], as well as each other in context of the questions defined in Section 5.2.2.

5.4.1 Comparison with Popular Embedding Methods

Due to difficulties in reproducing the exact testing environment, e.g. not being able to use the complete text corpus, the results are not always comparable to those in [32]. However, it is quite safe to say that the Com2Sem model as configured here generally performs worse.

Whereas most examined embeddings achieved a correlation of between 70 and 80 percent on the WS-353-SIM dataset, ngram2vec scoring best with 81.5% [32, Table II], the best tested Com2Sem model reached a correlation of 33.25%. Yet, since the notions of similarity required of the dataset and the Semantic Space differ considerably in parts, the results could also be limited by the categorization itself. Furthermore, it is noteworthy that the similarity task delivers a clear correlation at all while the categorization is still not optimal.

The performance of all embeddings on the BM dataset is not particularly high, about 45% [32, Table IV]. As this denotes a cluster purity metric, while in this thesis accuracy of the prediction task was measured, we cannot compare both results. Still, it is visible in Figure 5.5 and Appendix A.3 that some categories suffer from underfitting, especially those containing many training data if a low tree depth is chosen. Accordingly, it is advisable to only observe the results of models with a maximum tree depth greater than 1: there, we achieve 13.48% accuracy at maximum. Furthermore, in Appendix A.3 we notice that some

categories seem far more suitable for the prediction than others: for "country", accuracies of about 20% are achieved consistently and for "chemical element" scores over 30%.

The performance in Outlier Detection seems inconsistent compared with [32, Table V]: there, the GloVe embeddings achieved the best score with 15.09% accuracy and an OPP of 85.74% on the WikiSem-500 dataset. The best Com2Sem model scored 35.77% accuracy and an OPP of 63.90% – however, in this thesis it was not possible to test on the full dataset, so these results are not comparable either. Observed independently, the performance provides weak evidence that the Com2Sem model is at least capable of outlier detection.

As a consequence, the question whether Com2Sem shows advantages over popular embedding models on intrinsic NLP tasks, can be answered with "no" in general. Yet, since the model does achieve positive results, we conclude that there are concepts of human-perception that can be described by a small amount of context word combinations more or less reliably. That is also supported by the good results in a similarity task achieved by reduced co-occurrence vectors displayed in Figure 5.3.

5.4.2 Summary

One problem of the model in all tasks is that it reacts quite drastically to changes: even re-ordering of the co-occurrence features or small parameter changes (e.g. increasing or decreasing the maximum depth slightly) can lead to vastly different results, even if the random seed is fixed. A potential solution could be to not use single decision trees but random forests [15] – which, however, would complicate following the explainability principle.

In the Concept Categorization tasks it showed that the choice of features – and related to that any hierarchical structure – should be considered well: while in Section 5.3.2 countries, languages and, to a restricted degree, some other categories have been separated relatively accurately for especially large window sizes, it is problematic for loosely related word classes like "adjectives for people". It is noteworthy that COUNTRY belongs to the features best predicted in Appendix A.3 as well.

We as such relax the original hypothesis: There *are* human-understandable concepts for which a small number of word contexts is sufficient to separate them from other concepts. It might also be possible to describe some concepts by changing the co-occurrence space, for example by building a co-occurrence matrix not on the context *words* themselves but their category predicted by another model; applied to the "adjectives for people" example: words that often precede a word classified as HUMAN, but not with the same frequency as articles, are predicted as "adjectives for people".

Comparing the performances in Word Similarity and Concept Categorization, they seem to behave nearly anti-proportional referring to the window size: as the classification accuracy generally increases with the window size, the word similarity correlation mostly decreases. The same phenomenon is visible in Figure 5.6 for outlier detection, the compactness score of which is again based on similarity of vectors. A hypothesis already mentioned in Section 5.3.1 is that for larger windows the trees gain more information to avoid underfitting and as such manage to construct conditions better rejecting unsuitable words. Since these conditions are partly too strict, as shown by the example of low accuracies in Figure 5.4, more and more words are rejected when advancing in the tree structure. Accordingly, for applicability in similitary tasks it seems to be more suitable if trees underfit, which is the opposite to what would be advisable for categorization. Both problems could be solved if only the trees' accuracy is enhanced without introducing more false-positives.

Therefore, the Semantic Space seems to not yet have an optimal structure: features near the top-level are not sufficiently visible in the English language to be figured out by a decision tree reliably. Accordingly, the high-level features currently contained in Object should be moved to the top-most level. For example, Potag and Geographical could both be top-level elements, and concepts like Furniture or Informative are in any case no plausible sub-concepts of Instrument, from a language-related point of view (both scoring near-zero accuracy in Appendix A.3). Musical Instrument as opposed to that does achieve comparatively reasonable results as a sub-concept of Instrument already.

Chapter 6

Conclusion

6.1 Conclusion

This paper proposed a novel approach to learn word embeddings in a supervised manner, with both construction and prediction being interpretable, explainable and transparent. Hereby, decision trees are trained in order to find a number of context words that only occur regularly with words defining a specific semantic or grammatical property.

To evaluate whether the approach is applicable, the Com2Sem model has been introduced, which predicts embedding vectors in a hierarchical space from co-occurrence vectors. The model is even capable of predicting multiple embeddings per word, thus supporting homonymy detection. Two target spaces designed specifically for evaluation purposes were presented in Section 3.3.

Com2Sem has been implemented in Python, making use of some popular libraries for Data Science and Natural Language Processing, such as Scikit-learn, SciPy, NLTK and NumPy. The implementation includes a module for co-occurrence matrix construction as well as the Training Data Creator, a Tkinter-based GUI to design and export training data for the Com2Sem model.[1]

The results achieved in the evaluation of three intrinsic evaluators for word embeddings provide evidence that the basic approach is functional, even though the results are not optimal compared with popular word embedding models. We conclude that there are in fact semantic concepts that can be recognized in natural language by observing only a small set of context words, on the other hand we also learned that not all semantic categories can be predicted equally well on pure text corpora.

[1] The implementation can be found on Github at `https://github.com/NeclusMandarius/com2sem`.

However, an experiment conducted for word similarity tasks suggests another application for the model: context words selected by the decision trees appear to contain enough information to achieve correlations similar to those of the complete co-occurrence vectors. Therefore, the approach might offer a new transparent way of reducing the dimensionality of distributional spaces without negatively affecting the performance in NLP tasks.

Even though the reduced co-occurrence vectors scored better in the evaluation than embeddings, the *normalized concept overlapping* – introduced to measure similarity of embeddings as defined in Section 2.5 – shows a positive correlation as well, scoring similarly high in some cases. Accordingly, we conclude that the score function is generally suitable for its intended purpose.

Apart from the actual proposal, different techniques of how to collect co-occurrence matrices have been discussed in Section 3.2.2, relating to both technical and conceptional properties. These choices do not only result in practical limitations such as memory-demand, but influence the model's success in NLP tasks as well, as it showed in Chapter 5. While tasks based on similarity perform better on small window sizes which require the model to generate more general trees, the opposite holds for word categorization.

6.2 Outlook

Since a new approach was introduced and this thesis could not examine all its implications and properties, there are several open questions which might be interesting for future work.

In the evaluation, it turned out that the Semantic Space is not yet ideally structured, as its low-level features could not be trained optimally – this suggests that their semantical concepts are not sufficiently visible in the English language and thus lead to either too specialized or too generalized trees. Accordingly, the Semantic Space should be redefined to better represent actual language structures.

Related to this, it could be interesting to analyze whether there are ways to automatically generate an optimally structured target space for some specific objective. Given a dataset containing a variety of different meaning aspects for a lot of words, even an evolutionary approach would be imaginable, by nesting and combining space structures to maximize accuracy and minimize the false-positive rate. Since the space structure then models real language properties, information can be gained also from a linguistic point of view.

Creating a suitable target space, whether automatically or by manual design, would also be essential to test Com2Sem in downstream NLP tasks, which have not been conducted in this thesis, since intrinsic evaluators are better suited for a general overview. Still, adapting the approach for real-life applications is an important next step to measure its practicality.

On a theoretical note, a relevant subject for research could be how the decision tree based models are related to popular neural embeddings: if some properties of neural embeddings could be modeled by Com2Sem, for which it is possible to show justifications and graphical representations, one could potentially gain more insight into their inner structure.

To suit this purpose, the proposed model itself should also be improved. One prominent problem is that decision trees are highly randomized in practice (as cited in Section 2.4.2) which can lead to notable differences in prediction even when changing parameters or training data only slightly. This has also been observed in the course of evaluation. A possible solution would be to use random forests instead of decision trees, however, potentially introducing a trade-off between accuracy and explainability.

Relating to parameter changes, it turned out that the split criterion's choice has relevant and mostly consistent effects on the results of concept categorization. While entropy shows advantages over gini for low tree depths, the opposite holds for high tree depths, even though less strongly. In [16] it is stated that the gini and the entropy metric agree in most cases, so it could be a candidate topic for research to analyze why that appears not to apply to co-occurrence matrices.

As a final note, this thesis bases its evaluation heavily on classifier features. For further work on the Com2Sem model, a detailed analysis of regression features still needs to be conducted.

Appendix A

Appendix

A.1 Addition to the Semantic Space: Complete Structure

The Semantic Space consists of 172 features, i.e. dimensions. The structure presented here is separated into several figures to provide a better overview. Classifier features are underlined for clear differentiation from regression features.

The features are provided a short explanation of their meaning (if necessary) and some positive examples in the training data if it is given any exclusively (i.e. samples for child features are ignored). In the case of regression, one word from each end of the scala is shown.

The space has been constructed by the example of MultiNet sorts and features [13], providing a representation of semantics as a graph. The training data does also contain many example words named in the paper cited above.

- POSITIVITY: whether a concept has some positive connotation to it in general
  0: virus, 1: friend
- OBJECT: → Figure A.2
- SITUATION: → Figure A.13
- SITUATIONAL DESCRIPTOR: → Figure A.14
- QUALITY: → Figure A.15
- QUANTITY: → Figure A.16
- GRADUATOR: concepts only defining relativizing context for other concepts
  - QUALITATIVE GRADUATOR: the graduator operates on total qualities (very, especially, ...)
  - QUANTITATIVE GRADUATOR: the graduator operates on quantities (almost, approximately, ...)

Figure A.1: Top-level structure of the Semantic Space.

OBJECT: concepts perceived as an entity as opposed to situations or qualities

- INDIVIDUAL: whether a concept describes an individual or a group
  0: crowd, 1: author
- SPATIAL: whether a concept is perceived as being grounded in space
  0: birthday, 1: height
- PART OF: whether a concept is usually perceived as a part of some other entity
  0: rose, 1: blossom
- INFO: whether a concept provides additional information
  0: rabbit, 1: book
- FICTIONAL: whether a concept is fictional or grounded in the real-world
  0: country, 1: dragon
- CONCRETE OBJECT: → Figure A.3
- ABSTRACT OBJECT: → Figure A.10
- VIRTUAL OBJECT: → Figure A.12

Figure A.2: OBJECT subtree of the Semantic Space

CONCRETE OBJECT: entities representing spatial objects

- ARTIFICIAL: whether an object developed naturally or is constructed by humans
  0: ore, 1: violin
- AXIAL: whether an object possesses a spatial axis
  0: ball, 1: ladder
- MOVABLE: whether an object is movable (by itself or others)
  0: mountain, 1: desk
- INSTRUMENTAL: whether an object is usually perceived as a tool
  0: mammal, 1: basket
- CONTAINER: whether an object can serve as a container for other objects
  0: hammer, 1: cupboard
- SPECIFIC: whether a specific entity is pointed out or the concept describes a class of objects
  0: city, 1: Berlin
- DISCRETE OBJECT: → Figure A.4
- SUBSTANCE: → Figure A.9

Figure A.3: CONCRETE OBJECT subtree of the Semantic Space.

DISCRETE OBJECT: countable entities (at least entities used as if countable)

- CLOTHES: objects used as clothing
  - OUTDOOR: whether the clothing is usually worn outdoors
    0: –, 1: jacket
  - UNDERWEAR: whether the clothing is used as underwear
    0: jacket, 0.35: sock
  - ALWAYS: whether the clothing can be worn in all situations (pant, sock, ...)
  - WINTER: whether the clothing is worn only if it is cold (coat, glove, ...)
  - SUMMER: whether the clothing is worn only during good weather (blouse, slipper, ...)
  - RAIN: whether the clothing is worn only if it is rainy/stormy
- BUILDING (hospital, skyscraper, ...)
- GEOGRAPHICAL: → Figure A.5
- INSTRUMENT: → Figure A.6
- POTAG: → Figure A.7

Figure A.4: DISCRETE OBJECT subtree of the Semantic Space.

GEOGRAPHICAL: concrete or conceptional entities with a geographic location

- CARRIES WATER: an entity carrying water
  0: park, 1: river
- LANDSCAPE: concepts describing a wide landscape (field, lake, ...)
- LANDSCAPE FEATURE: concepts describing elements of a landscape viewed as single entities (canyon, hill, ...)
- BIOME: concepts describing a full biome (desert, ocean, ...)
- CONSTRUCTED: geographical entities introduced or built by humans
  - BIG: whether the entity is perceived as large or small
    0.3: Venezuela, 1: China
  - ICONIC BUILDING: the entity denotes or is the name of some kind of building perceivable as landmark (castle, pyramid, ...)
  - CITY: the entity denotes or is the name of a city (Berlin, Paris, ...)
  - COUNTRY: the entity denotes or is the name of a country or district (country, England, ...)

Figure A.5: GEOGRAPHICAL subtree of the Semantic Space.

INSTRUMENT: objects not being potential agents used as tools

- SIZE: whether the object is large or rather small
  0: triangle, 1: container
- MUSICAL INSTRUMENT: tools used as musical instrument (accordion, banjo, ...)
- SPORT DEVICE: objects used in sports (football, basket, ...)
- SCIENTIFIC DEVICE: instruments used in science (barometer, telescope, ...)
- DEVICE: other devices (hammer, tool, ...)
  - MECHANICAL: whether the device works mechanically
    0: thermometer, 1: nail
  - WEAPON: the object is used as a weapon (gun, missile, ...)
  - MEDICAL: the tool is used for medical purposes (needle, thermometer, ...)
- CONTAINING: the entity can act as a container for other objects (bag, box, ...)
- FURNITURE: objects used as furniture (chair, table, ...)
- INFORMATIVE: objects containing information (article, book, ...)

Figure A.6: INSTRUMENT subtree of the Semantic Space.

Potag: potential agent of an action

- Animate: the agent is animate
  - Microscopic: microscopical beings (bacterium, virus, ...)
  - Animal
    - Living on Land: whether the animal usually lives on land
      0: fish, 1: sheep
    - Living in Sea: whether the animal usually lives in water
      0: rabbit, 1: shark
    - Flies: whether the animal can fly
      0: penguin, 1: owl
    - Mammal (fox, dolphin, ...)
    - Reptile (snake, lizard, ...)
    - Bird (eagle, kingfisher, ...)
    - Fish (fish, shark, ...)
    - Amphibian (frog, salamander, ...)
    - Insect (butterfly, hornet, ...)
    - Arachnid (spider, scorpion, ...)
  - plant
    - Fruit: fruits and vegetables (apple, tomato, ...)
    - Tree (oak, willow, ...)
    - Flower (tulip, rose, ...)
    - Shrub (bush, ivy, ...)
    - Weed (grass, moss, ...)
    - Mushroom (mushroom, ...)
  - Human
    - Family Related: the word is related to family
      0: judge, 1: child
    - Profession (farmer, priest, ...)
    - Title (chief, king, ...)
    - State: humans in a specific natural or situational state (child, student, ...)
    - Group (family, people, ...)
- Inanimate: → Figure A.8

Figure A.7: Potag subtree of the Semantic Space

Inanimate: the agent is inanimate

- Digital: whether/how much the tool works digitally
  0: grill, 1: robot
- Transport (bus, car, ...)
- Calculation: tools for computation (computer, calculator, ...)
- Industry: industrial machines (robot, motor, ...)
- Home: electricity-powered tools of everyday life (microwave, oven, ...)
- Military: machines or weapons with military uses (warship, tank, ...)

Figure A.8: Inanimate subtree of the Semantic Space.

Substance: uncountable concrete objects

- Food: whether the object is usually eaten
  0: ethanol, 1: juice
- Transparent: whether the material is transparent
  0: iron, 1: water
- Radioactive: whether the substance is radioactive
  0: oil, 1: –
- Medicine: whether the substance is used as medicine
  0: water, 1: –
- Poisonous: whether the substance is usually poisonous
  0: tea, 1: poison
- Fluid: the object's state of matter is fluid (water, oil, ...)
- Firm: the object's state of matter is firm (concrete, quartz, ...)
  - Metal: the substance is metallic (gold, iron, ...)
  - Organic: the substance is organic (wood, soil, ...)
- Gas: the object's state of matter is gas (air, hydrogen, ...)

Figure A.9: Substance subtree of the Semantic Space.

Abstract Object:

- Methodical: whether the concept describes a method, technique or algorithm
  0: charm, 1: algorithm
- Discipline: the concept is related to some professional or scientific discipline
  - Art: the concept is related to art
    - Music: the concept is related to music (jazz, hymn, ...)
    - Painting: the concept is related to visual art (painting, modernism, ...)
    - Creative Writing: the concept is related to writing or literature (storytelling, plot, ...)
  - Science: the concept is related to science (biology, chemistry, ...)
  - Sports: the concept is related to sports (rugby, tennis, ...)
- attribute: the word denotes the abstract concept of an attribute that can be connected with other entities
  - Measurable Attribute: the attribute is measurable (height, weight, ...)
  - Non-Measurable Attribute: the attribute is absolute (form, charm, ...)
- Language: concepts related to language, including language names (German, Indonesian, ...)
- Relationship: the concept represents a relation between different entities (causality, similarity, ...)
- Ideal Object: the concept represents an idealized abstraction of some entity or situation (religion, justice, ...)
- Abstract Temporal Object: the concept denotes an object corresponding to a time period or date (Renaissance, birthday, ...)
- Modality: words conditioning or motivating the execution or outcome of a situation (necessity, intention, ...)
- Situational Object: concept describing a situation as object
  - Dynamic Situational Object: object describing a dynamic situation (race, change, ...)
  - Static Situational Object: object describing a state of being (sleep, equilibrium)
- Institution: → Figure A.11

Figure A.10: Abstract Object subtree of the Semantic Space.

INSTITUTION: words denoting legal institutions or communities

- INSTITUTION SIZE: on which scale a respective institution usually operates
  0: administration, 1: UNO
- LOCAL INSTITUTION: institution responsible for a rather small environment (administration, board, ...)
- CULTURAL INSTITUTION: institution concerned with cultural activities (club, pub, ...)
- NATIONAL INSTITUTION: institution operating on a national scope (ministry, government, ...)
- RELIGIOUS INSTITUTION: some sort of church or institution concerned with religion (church, cult, ...)
- GLOBAL INSTITUTION: institution operating on a global scope (UNO, UNESCO, ...)
- ECONOMICAL INSTITUTION: institution with economical interests (enterprise, company, ...)

Figure A.11: INSTITUTION subtree of the Semantic Space.

VIRTUAL OBJECT: the concept appears in the context of computer science or the virtual world

- OPERATING SYSTEM: the word is the name of or related to operating systems (OS, Linux, ...)
- PROGRAM: the word is the name of or related to computer programs (excel, browser, ...)
- DATA: the concept appears as synonym or some kind of data (data, information, ...)
- DATA STRUCTURE: the concept denotes some structural organization of data (list, map, ...)
- DATA TRANSMITTER: the concept denotes technology, protocols or other techniques transmitting data (internet, LAN, ...)
- USER INTERFACE: the concept relates to digital user interfaces (button, UI, ...)
- EXTERNAL INTERFACE: the concept relates to interfaces for computers

Figure A.12: VIRTUAL OBJECT subtree of the Semantic Space.

Situation: concepts perceived as a state, situation or action

- Dynamic Situation: situation with participants changing states
  - Action: situation actively carried out by one participant
    - Reflexive Action: action affecting the actor
      0: write, 1: go
    - Transitive Action: action requiring an object
      0: go, 1: write
    - Animate Action: action only or usually performed by animate beings
      - Human Action: action only or usually performed by humans (write, sell, ...)
      - Animal Action: action only or usually carried out by animals (bite, clean, ...)
      - Plant Action: action only or usually seen with plants (photosynthesis, root, ...)
    - Inanimate Action: action performed by e.g. machines (rotate, lift, ...)
  - Happening: dynamic situation not actively sustained by any participant
    - Natural Happening: situation with natural cause (rain, snow, ...)
    - Supernatural Happening: situation observed as supernatural (miracle, mystery, ...)
    - Random Happening: situation with no obvious cause (encounter, happening, ...)
- Static Situation: situation in which participants mostly remain in their initial state
  - Animate Situation: situation only or usually found with animate beings
    - Human Situation: situations only or usually found with humans (stand, read, ...)
    - Animal Situation: situations semantically mainly connected to animals (hibernation, mating, ...)
    - Plant Situation: situations connected to plants
  - Inanimate Situation: situations in which inanimate objects can be found (running, off, ...)

Figure A.13: Situation subtree of the Semantic Space.

Situational Descriptor: concepts granting additional information like time or modality with respect to a situation

- Time: concept specifying a time-related information
  - Time Relation: concept specifying a relation in time (past, future, ...)
  - Duration (duration, minute, ...)
  - Point in Time (yesterday, Monday, ...)
- Modal Situational Descriptor: concept describing a situation on a level of conditionality or motivation

Figure A.14: Situational Descriptor subtree of the Semantic Space.

Quality: concepts defining qualities of an entity, e.g. colors

- Property: describes a direct property of an entity
  - Total Quality: qualities that either apply or do not apply (empty, silent, ...)
  - Gradable Quality: qualities that may be applicable partly or in different degrees
    - Measurable Quality: the property can be measured by some means (green, small, ...)
    - Non-Measurable Quality: the property is not measurable (friendly, heroic, ...)
- Relational Quality: the quality is described by means of a relation between entities (equivalent, similar, ...)
- Functional Quality: the quality describes an entity's contextual relation
  - Operational Quality: describes an entity in some relational context (fourth, last, ...)
  - Associative Quality: describes an entity by contextual associations (physical, philosophical, ...)

Figure A.15: Quality subtree of the Semantic Space.

Quantity: numbers or units of measurement

- Quantificator: denotes some concept that can be used to count, measure or estimate
  - Numerical Quantificator: words expressing numbers or specific values (one, 2, ...)
  - Non-Numerical Quantificator: quantifiers not denoting specific values (all, several, ...)
- Unit of Measurement (kg, meter, ...)
- Measurement: quantifiers along with a unit of measurement

Figure A.16: Quantity subtree of the Semantic Space.

A.2 Addition to COM Construction: Example

The NLTK Python package can be used to quickly build a co-occurrence matrix and calculating PMI values of it:

```
1  from nltk import collocations
2  from nltk.tokenize import word_tokenize
3
4  win_size = 2
5
6  with open("our-corpus.txt", encoding="utf-8") as file:
7          corpus = word_tokenize(file.read())
8          # still includes punctuation, stopwords etc.
9
10 # Construct the BigramCollocationFinder,
11 # the actual co-occurrence "matrix"
12 bcf = collocations.BigramCollocationFinder.from_words(corpus, win_size)
13 # Using the static from_words method, the window is
14 # left-aligned: for win_size = 2, if only "X Y Z" is contained in
15 # the corpus, ("X", "Y", 1) and ("Y", "Z", 1) are contained in bcf,
16 # but NOT ("Y", "X", 1) or ("Z", "Y", 1).
17
18 bam = collocations.BigramAssocMeasures()
19
20 # Apply PPMI as a filter (reduces memory usage)
21 bcf.apply_ngram_filter(
22         lambda w1, w2: (bcf.score_ngram(bam.pmi, w1, w2) or 0) <= 0
23 )
24
25 # Note: bcf.score_ngram(_metric_, w1, w2) returns None if
26 # the word combination is not contained.
```

Along with the Com2Sem model implementation, a wrapper class for this use of the `nltk.collocations.BigramCollocationFinder` has been created which offers some additional functionality.

A.3 Addition to Concept Categorization

The following table displays the mapping of BM concepts to Semantic Space features, along with the number of samples occurring in both cluster and training data (3rd column) and

the cluster-specific accuracies of four experiments. In the header of the last four, the first number determines the window size and the second the maximum tree depth. The accuracy is specified in percent.

Note that high accuracies especially for models with depth 1 are less reliable, since they could be produced by unbalanced trees predicting the specific category by default.

BM concept	Semantic Space feature	#	2 (1)	2 (15)	3 (1)	3 (15)
precious stone	firm	3	0.00	3.70	0.00	7.41
unit of time	unit of measurement	0	2.94	2.94	2.94	2.94
relative	family related	2	2.56	2.56	7.69	5.13
unit of distance	unit of measurement	3	6.67	6.67	30.00	6.67
metal	metal	3	0.00	0.00	22.73	22.73
type of reading material	informative	4	0.00	1.28	2.56	0.00
military title	title	0	23.08	19.23	26.92	11.54
four-footed animal	mammal	5	8.97	7.69	8.97	6.41
kind of cloth	organic	0	0.00	2.74	9.59	16.44
color	quality	1	23.91	10.87	73.91	10.87
kitchen utensil	instrument	1	75.81	12.90	40.32	12.90
bldg for religious service	building	1	36.36	12.12	12.12	0.00
part of speech	discipline	0	1.47	10.29	11.76	5.88
article of furniture	furniture	3	3.51	0.00	3.51	0.00
part of the human body	animate	0	66.3	13.04	44.57	31.52
fruit	fruit	1	5.88	0.00	2.94	5.88
weapon	weapon	3	1.18	0.00	15.29	3.53
elective office	human	4	40.54	43.24	37.84	24.32
type of human dwelling	building	2	18.64	8.47	5.08	1.69
alcoholic beverage	fluid	1	0.00	0.00	2.63	5.26
country	country	6	20.69	19.54	16.09	17.24
crime	human action	0	2.74	2.74	0.00	2.74
carpenters tool	instrument	3	75.00	10.00	33.33	15.00
member of the clergy	profession	1	26.32	10.53	18.42	7.89
substance to flavor food	substance	0	11.69	6.49	11.69	18.18
type of fuel	substance	3	29.41	17.65	31.37	31.37
occupation or profession	profession	6	25.17	14.97	8.84	14.97
natural earth formation	landscape feature	3	11.11	12.22	7.78	8.89
sport	sports	2	3.85	7.69	1.92	0.00
weather phenomenon	natural happening	3	4.92	1.64	0.00	4.92
article of clothing	clothes	9	28.57	17.46	26.98	23.81

part of a building	building	1	13.33	3.33	2.50	0.00
chemical element	substance	4	57.75	33.8	53.52	47.89
musical instrument	musical instrument	8	42.5	25.00	37.5	20.00
kind of money	unit of measurement	0	0.00	0.00	12.70	3.17
type of music	music	3	0.00	0.00	1.90	0.00
bird	bird	5	2.74	5.48	2.74	10.96
nonalcoholic beverage	fluid	4	5.88	0.00	5.88	11.76
type of vehicle	transport	3	5.97	4.48	10.45	4.48
science	science	2	2.27	2.27	20.45	9.09
toy	instrument	2	71.76	12.94	56.47	11.76
type of dance	discipline	0	0.00	3.17	1.59	1.59
vegetable	fruit	2	0.00	0.00	5.26	2.63
type of footgear	clothes	2	13.33	6.67	11.11	17.78
insect	insect	3	5.56	8.33	0.00	2.78
girls first name	human	0	1.47	6.99	51.84	19.49
males first name	human	0	10.29	16.87	67.49	27.98
flower	flower	2	1.89	1.89	1.89	0.00
disease	science	0	0.00	0.00	0.00	0.00
tree	tree	2	14.08	12.68	2.82	4.23
type of ship	transport	2	1.33	2.67	8.00	4.00
fish	fish	1	11.11	9.88	4.94	2.47
snake	reptile	3	4.88	0.00	2.44	2.44
city	city	4	10.42	8.33	25.69	14.58
state	country	2	23.68	39.47	5.26	31.58
college or university	constructed	0	2.3	3.45	20.69	40.23

A.4 Addition to Topic Space Evaluation

The following tables display the category-specific accuracies of eight of the experiments presented in Figure 5.6, those using the `"entropy"` criterion. In the header of the last four columns, the first number determines the window size and the second the maximum tree depth. The cells contain entries of the format “X / Y” that must be read “accuracy of X percent with Y false-positives”. The test data consists of 4,467 samples in total.

Category	2 (1)	2 (15)	3 (1)	3 (15)
adjectives for people	91.02 / 2,819	13.67 / 152	23.83 / 663	5.86 / 68
animal	68.02 / 2,090	15.38 / 112	12.15 / 28	12.55 / 143

art	83.65 / 3,353	3.85 / 78	17.31 / 263	2.88 / 129
astronomy	8.43 / 180	4.82 / 92	83.13 / 3,136	6.02 / 192
beach	1.69 / 64	6.78 / 165	20.34 / 782	3.39 / 132
birds	62.00 / 2,383	2.00 / 139	88.00 / 3,431	42.00 / 908
boat	91.15 / 3,704	5.31 / 131	86.73 / 3,290	3.54 / 214
body	14.74 / 18	9.47 / 97	86.32 / 3,058	9.47 / 127
buildings	94.78 / 3,104	11.30 / 227	66.96 / 2,447	9.57 / 118
carparts	82.46 / 3,543	12.28 / 386	22.81 / 192	15.79 / 166
castle	64.15 / 2,222	5.66 / 311	75.47 / 3,508	9.43 / 245
christmas	81.90 / 3,643	3.81 / 220	82.86 / 3,618	0.95 / 193
clothes	79.81 / 3,250	18.27 / 600	86.54 / 3,063	25.00 / 711
colors	43.48 / 1,714	7.25 / 146	8.70 / 71	10.14 / 182
computer	64.10 / 3,108	5.13 / 118	5.13 / 24	14.10 / 93
constitution	23.38 / 138	9.09 / 61	44.16 / 289	15.58 / 180
container	9.09 / 259	9.09 / 107	5.45 / 43	7.27 / 173
cooking	11.46 / 178	9.38 / 190	16.67 / 578	6.25 / 244
country	46.25 / 727	33.75 / 42	97.50 / 1,122	58.75 / 44
dentist	76.83 / 3,072	3.66 / 178	68.29 / 3,436	2.44 / 170
doctor	78.23 / 3,568	3.40 / 77	76.87 / 3,277	10.88 / 196
driving	7.29 / 28	6.25 / 97	10.42 / 41	6.25 / 150
emotions	88.79 / 3,141	12.07 / 91	43.97 / 1,359	4.31 / 67
energy	25.33 / 202	8.00 / 168	68.00 / 3,133	16.00 / 193
family	12.75 / 223	3.92 / 101	21.57 / 288	13.73 / 126
farm	12.62 / 633	4.85 / 185	33.98 / 1,674	12.62 / 139
food	83.98 / 3,191	10.55 / 186	76.17 / 2,559	14.45 / 166
house	81.00 / 3,540	7.00 / 246	69.00 / 2,941	10.00 / 220
jobs	55.03 / 359	16.78 / 87	34.90 / 143	14.09 / 111
land forms	73.44 / 3,189	8.59 / 127	70.31 / 2,397	7.03 / 102
languages	52.63 / 42	57.89 / 83	52.63 / 54	49.47 / 117
leaders	45.79 / 403	8.41 / 89	27.10 / 62	24.30 / 122
legal	72.73 / 3,012	6.61 / 156	32.23 / 723	7.44 / 98
math	81.25 / 2,564	10.00 / 151	36.25 / 635	7.50 / 168
military	15.70 / 298	3.49 / 120	34.88 / 395	6.98 / 101
money	20.45 / 296	11.36 / 171	29.55 / 906	3.41 / 168
negative words	93.63 / 3,079	7.64 / 148	27.39 / 315	14.65 / 135
office	2.70 / 20	5.41 / 119	25.23 / 568	3.15 / 95
people	55.24 / 428	11.43 / 153	73.33 / 3,228	8.57 / 120

pirate	82.07 /3,505	4.83 / 137	82.07 / 3,451	9.66 / 241
plants	84.09 / 3,721	3.41 / 114	89.77 / 3,594	9.09 / 199
positive words	88.11 / 2,852	10.13 / 154	25.99 / 597	2.20 / 67
restaurant	84.62 / 3,768	8.79 / 177	84.62 / 3,524	9.89 / 292
sports	6.93 / 13	15.35 / 266	8.91 / 23	7.43 / 106
transportation	63.44 / 1,811	4.59 / 136	75.27 / 2,887	15.05 / 141
vacation	7.34 / 58	4.59 / 136	8.26 / 85	6.42 / 174
virtues	82.56 / 2,339	13.95 / 155	86.05 / 2,103	25.58 / 176
weather	88.03 / 3,408	15.38 / 233	7.69 / 67	10.26 / 176

Category	7 (1)	7 (15)	21 (1)	21 (15)
adjectives for people	14.84 / 398	4.69 / 66	75.39 / 2,952	5.47 / 169
animal	25.10 / 73	14.57 / 132	63.16 / 272	26.72 / 122
art	21.15 / 157	16.35 / 93	46.15 / 165	25.00 / 269
astronomy	49.40 / 334	9.64 / 65	78.31 / 272	42.17 / 83
beach	15.25 / 157	6.78 / 220	37.29 / 405	28.81 / 373
birds	59.00 / 1586	3.00 / 175	69.00 / 154	48.00 / 141
boat	27.43 / 142	24.78 / 136	64.60 / 233	23.89 / 84
body	17.89 / 82	13.68 / 85	64.21 / 302	32.63 / 302
buildings	33.04 / 137	10.43 / 113	31.30 / 94	18.26 / 302
carparts	47.37 / 1,781	12.28 / 109	54.39 / 474	21.05 / 379
castle	52.83 / 2,258	7.55 / 198	58.49 / 480	26.42 / 157
christmas	74.29 / 3,054	3.81 / 181	40.00 / 532	26.67 / 310
clothes	76.92 / 3,042	9.62 / 152	73.08 / 2,437	33.65 / 257
colors	39.13 / 232	26.09 / 178	69.57 / 424	24.64 / 114
computer	44.23 / 305	10.26 / 68	72.44 / 311	36.54 / 61
constitution	59.74 / 367	18.18 / 112	84.42 / 517	25.97 / 64
container	9.09 / 150	7.27 / 180	29.09 / 236	9.09 / 90
cooking	78.13 / 3,332	0.00 / 108	75.00 /3,083	4.17 / 131
country	57.50 / 144	52.50 / 31	62.50 / 108	32.5 / 18
dentist	71.95 / 3,317	4.88 / 394	46.34 / 216	9.76 / 275
doctor	16.33 / 40	15.65 / 68	55.10 / 190	25.85 / 318
driving	29.17 / 177	7.29 / 138	60.42 / 610	16.67 / 353
emotions	68.97 / 2,240	14.66 / 180	16.38 / 194	12.93 / 131
energy	72.00 / 2,754	26.67 / 182	53.33 / 122	38.67 / 82
family	63.73 / 1,506	12.75 / 149	52.94 / 410	23.53 / 345
farm	23.30 / 95	10.68 / 85	41.75 / 497	28.16 / 349

food	40.23 / 582	12.50 / 152	57.81 / 295	30.86 / 282
house	4.00 / 139	3.00 / 145	27.00 / 180	21.00 / 307
jobs	54.36 / 1,327	10.74 / 111	32.21 / 756	6.71 / 108
land forms	62.50 / 2,337	7.03 / 91	79.69 / 1,645	14.84 / 119
languages	66.32 / 88	61.05 / 49	91.58 / 180	57.89 / 87
leaders	42.06 / 85	14.95 / 73	59.81 / 324	22.43 / 79
legal	39.67 / 175	5.79 / 157	67.77 / 285	24.79 / 74
math	40.00 / 443	13.75 / 114	71.25 / 509	17.50 / 320
military	49.42 / 513	15.12 / 179	41.86 / 268	20.93 / 319
money	63.64 / 346	23.86 / 116	59.09 / 578	23.86 / 334
negative words	81.53 / 2,945	7.64 / 145	64.33 / 2,194	13.38 / 189
office	31.98 / 517	9.01 / 164	48.65 / 460	16.67 / 289
people	5.71 / 139	8.57 / 99	71.43 / 1,306	4.76 / 115
pirate	22.76 / 293	11.72 / 192	40.00 / 396	17.93 / 339
plants	80.68 / 3,341	3.41 / 122	79.55 / 858	6.82 / 131
positive words	61.23 / 1,793	12.33 / 141	39.21 / 997	9.25 / 148
restaurant	28.57 / 156	16.48 / 121	15.38 / 156	12.09 / 118
sports	54.46 / 1,728	11.88 / 129	41.09 / 559	14.85 / 49
transportation	75.27 / 2,693	9.68 / 105	53.76 / 418	18.28 / 92
vacation	6.42 / 137	8.26 / 106	50.46 / 656	15.60 / 305
virtues	80.23 / 2,302	16.28 / 87	80.23 / 1,524	19.76 / 142
weather	23.93 / 177	11.97 / 115	46.15 / 249	23.08 / 105

List of Figures

List of Algorithms

Bibliography

[1] *1.10. Decision Trees — scikit-learn 0.24.0 documentation*, 2021. [Online; accessed 04-January-2021].

[2] *sklearn.tree.DecisionTreeRegressor — scikit-learn 0.24.1 documentation*, 2021. [Online; accessed 25-February-2021].

[3] Agirre, Eneko and Alfonseca, Enrique and Hall, Keith and Kravalova, Jana and Paşca, Marius and Soroa, Aitor: *A Study on Similarity and Relatedness Using Distributional and WordNet-based Approaches.* In *Proceedings of Human Language Technologies: The 2009 Annual Conference of the North American Chapter of the Association for Computational Linguistics*, pages 19–27, Boulder, Colorado, June 2009. Association for Computational Linguistics.

[4] Bolukbasi, Tolga and Chang, Kai-Wei and Zou, James and Saligrama, Venkatesh and Kalai, Adam: *Man is to Computer Programmer as Woman is to Homemaker? Debiasing Word Embeddings.* In *Proceedings of the 30th International Conference on Neural Information Processing Systems*, NIPS'16, page 4356–4364, Red Hook, NY, USA, 2016. Curran Associates Inc.

[5] Breiman, L., J. Friedman, R. Olshen and C. J. Stone: *Classification and Regression Trees.* Wadsworth, 1984.

[6] Camacho-Collados, José and Navigli, Roberto: *Find the word that does not belong: A Framework for an Intrinsic Evaluation of Word Vector Representations.* In *Proceedings of the 1st Workshop on Evaluating Vector-Space Representations for NLP*, pages 43–50, Berlin, Germany, August 2016. Association for Computational Linguistics.

[7] Došilović, F. K., M. Brčić and N. Hlupić: *Explainable artificial intelligence: A survey.* In *2018 41st International Convention on Information and Communication Technology, Electronics and Microelectronics (MIPRO)*, pages 0210–0215, 2018.

[8] Fagarasan, Luana, Eva Maria Vecchi and Stephen Clark: *From distributional semantics to feature norms: grounding semantic models in human perceptual*

data. In *Proceedings of the 11th International Conference on Computational Semantics*, pages 52–57, London, UK, April 2015. Association for Computational Linguistics.

[9] Finkelstein, Lev, Evgeniy Gabrilovich, Yossi Matias, Ehud Rivlin, Zach Solan, Gadi Wolfman and Eytan Ruppin: *Placing Search in Context: The Concept Revisited*. In *Proceedings of the 10th International Conference on World Wide Web*, WWW '01, page 406–414, New York, NY, USA, 2001. Association for Computing Machinery.

[10] H. Hotelling: *Analysis of a complex of statistical variables into principal components*. Journal of Educational Psychology, 24:498–520, 1933.

[11] Harris, Charles R., K. Jarrod Millman, Stéfan J. van der Walt, Ralf Gommers, Pauli Virtanen, David Cournapeau, Eric Wieser, Julian Taylor, Sebastian Berg, Nathaniel J. Smith, Robert Kern, Matti Picus, Stephan Hoyer, Marten H. van Kerkwijk, Matthew Brett, Allan Haldane, Jaime Fernández del Río, Mark Wiebe, Pearu Peterson, Pierre Gérard-Marchant, Kevin Sheppard, Tyler Reddy, Warren Weckesser, Hameer Abbasi, Christoph Gohlke and Travis E. Oliphant: *Array programming with NumPy*. Nature, 585(7825):357–362, Sep 2020.

[12] Harris, Zellig S.: *Distributional Structure*. WORD, 10:146–162, 1954.

[13] Helbig, Hermann: *Knowledge Representation and the Semantics of Natural Language*, volume 9. 01 2006.

[14] Herbelot, Aurélie and Eva Maria Vecchi: *Building a shared world: mapping distributional to model-theoretic semantic spaces*. In *Proceedings of the 2015 Conference on Empirical Methods in Natural Language Processing*, pages 22–32, Lisbon, Portugal, September 2015. Association for Computational Linguistics.

[15] Ho, Tin Kam: *Random Decision Forests*. In *Proceedings of the Third International Conference on Document Analysis and Recognition (Volume 1) - Volume 1*, ICDAR '95, page 278, USA, 1995. IEEE Computer Society.

[16] Laura Elena Raileanu and K. Stoffel: *Theoretical Comparison between the Gini Index and Information Gain Criteria*. Annals of Mathematics and Artificial Intelligence, 41:77–93, 2004.

[17] Levy, Omer and Yoav Goldberg: *Neural Word Embedding as Implicit Matrix Factorization*. In *Proceedings of the 27th International Conference on Neural Information Processing Systems - Volume 2*, NIPS'14, page 2177–2185, Cambridge, MA, USA, 2014. MIT Press.

[18] Loper, Edward and Steven Bird: *NLTK: The Natural Language Toolkit*. CoRR, cs.CL/0205028, 2002.

[19] McRae, Ken, George S. Cree, Mark S. Seidenberg and Chris Mcnorgan: *Semantic feature production norms for a large set of living and nonliving things*. Behavior Research Methods, 37(4):547–559, Nov 2005.

[20] Mikolov, Tomas, Kai Chen, Greg Corrado and Jeffrey Dean: *Efficient Estimation of Word Representations in Vector Space*, 2013.

[21] Mikolov, Tomas, Quoc V. Le and Ilya Sutskever: *Exploiting Similarities among Languages for Machine Translation*, 2013.

[22] Miller, George A.: *WordNet: A Lexical Database for English*. Commun. ACM, 38(11):39–41, November 1995.

[23] Nalisnick, Eric, Bhaskar Mitra, Nick Craswell and Rich Caruana: *Improving Document Ranking with Dual Word Embeddings*. In *Proceedings of the 25th International Conference Companion on World Wide Web*, WWW '16 Companion, page 83–84, Republic and Canton of Geneva, CHE, 2016. International World Wide Web Conferences Steering Committee.

[24] Pearson, Karl: *LIII. On lines and planes of closest fit to systems of points in space*, November 1901.

[25] Pedregosa, F., G. Varoquaux, A. Gramfort, V. Michel, B. Thirion, O. Grisel, M. Blondel, P. Prettenhofer, R. Weiss, V. Dubourg, J. Vanderplas, A. Passos, D. Cournapeau, M. Brucher, M. Perrot and E. Duchesnay: *Scikit-learn: Machine Learning in Python*. Journal of Machine Learning Research, 12:2825–2830, 2011.

[26] Philip Blair and Yuval Merhav and Joel Barry: *Automated Generation of Multilingual Clusters for the Evaluation of Distributed Representations*, 2017.

[27] S. Deerwester and S. Dumais and T. Landauer and G. Furnas and R. Harshman: *Indexing by Latent Semantic Analysis*. J. Am. Soc. Inf. Sci., 41:391–407, 1990.

[28] Senel, Lutfi Kerem, Ihsan Utlu, Veysel Yücesoy, Aykut Koç and Tolga Çukur: *Semantic Structure and Interpretability of Word Embeddings*. CoRR, abs/1711.00331, 2017.

[29] Senel L. K., Utlu I., Yucesoy V. Koc A. Cukur T.: *Semantic Structure and Interpretability of Word Embeddings*. IEEE/ACM Trans. Audio, Speech and Lang. Proc., 26(10):1769–1779, October 2018.

[30] Virtanen, Pauli and Gommers, Ralf and Oliphant, Travis E. and Haberland, Matt and Reddy, Tyler and Cournapeau, David and Burovski, Evgeni and Peterson, Pearu and Weckesser, Warren and Bright, Jonathan and van der Walt, Stéfan J. and Brett, Matthew and Wilson, Joshua and Millman, K. Jarrod and Mayorov, Nikolay and Nelson, Andrew R. J. and Jones, Eric and Kern, Robert and Larson, Eric and Carey, C J and Polat, İlhan and Feng, Yu and Moore, Eric W. and VanderPlas, Jake and Laxalde, Denis and Perktold, Josef and Cimrman, Robert and Henriksen, Ian and Quintero, E. A. and Harris, Charles R. and Archibald, Anne M. and Ribeiro, Antônio H. and Pedregosa, Fabian and van Mulbregt, Paul and SciPy 1.0 Contributors: *SciPy 1.0: Fundamental Algorithms for Scientific Computing in Python.* Nature Methods, 17:261–272, 2020.

[31] W. F. Battig and W. Montague: *Category norms of verbal items in 56 categories A replication and extension of the Connecticut category norms.* Journal of Experimental Psychology, 80:1–46, 1969.

[32] Wang, Bin and Wang, Angela and Chen, Fenxiao and Wang, Yuncheng and Kuo, C.-C. Jay: *Evaluating Word Embedding Models: Methods and Experimental Results.* APSIPA Transactions on Signal and Information Processing, 8, 2019.