Fine-Grained Termhood Prediction for German Compound Terms Using Neural Networks

[†‡]**Anna Hätty** and [‡]**Sabine Schulte im Walde**
[†]Robert Bosch GmbH, Germany
[‡]Institute for Natural Language Processing, University of Stuttgart, Germany
anna.haetty@de.bosch.com, schulte@ims.uni-stuttgart.de

Abstract

Automatic term identification and investigating the understandability of terms in a specialized domain are often treated as two separate lines of research. We propose a combined approach for this matter, by defining fine-grained classes of termhood and framing a classification task. The classes reflect tiers of a term's association to a domain. The new setup is applied to German closed compounds as term candidates in the domain of cooking. For the prediction of the classes, we compare several neural network architectures and also take salient information about the compounds' components into account. We show that applying a similar class distinction to the compounds' components and propagating this information within the network improves the compound class prediction results.

1 Introduction

DOMAIN-SPECIFIC TERMS are linguistic expressions which characterize a domain, and the automatic recognition of terms is an important basis for further NLP tasks, such as thesaurus creation, automatic translation, and, more generally, for domain knowledge acquisition and for improving the comprehension of a domain. Automatic term recognition often comprises two steps: to evaluate a term candidate for its *unithood*, i.e. identifying boundaries of meaningful phrases, and then to evaluate its *termhood*, i.e. determining the degree of association to a specific domain (Kagueura and Umino, 1996). A related task to term recognition is the identification of domain terms unfamiliar to non-experts, i.e., identifying terms the average reader does not understand. This technology is mostly applied to the health domain, such as medical terms extracted to improve the communication between doctors and patients. The evaluated terms are mostly extracted from medical terminologies (Elhadad, 2006; Zeng-Treitler et al., 2008; Grabar and Hamon, 2014; Grabar et al., 2014). Since these terminologies are available, what constitutes a term can often be taken as a given. However, difficult terms occur in other domains as well, and huge terminologies are not always available. In this work, we make a first attempt to combine the two tasks of automatic term recognition and term difficulty identification by reformulating them as one problem considering different tiers of termhood. In our proposal, we define four classes of termhood, which range from general-language words to obscure, domain-specific terms. This replaces the concept of termhood treated as a binary problem often seen in term annotation (Arcan et al., 2014; Bernier-Colborne and Drouin, 2014; Zadeh and Handschuh, 2014a) and identification tasks (Ventura et al., 2014; Riedl and Biemann, 2015). We demonstrate the effectiveness of our tier system by first conducting an annotation task, followed by an automatic classification using neural networks tailored for our task.

In this work, we focus on the cooking domain and the German language. As a basis for the experiments, 400 German closed compounds are taken as term candidates. We focus on closed compounds for the following reasons: Closed compounds are complex expressions that consist of two or more simple words and contain no spaces or hyphens, e.g. *Meeresfrüchte* "seafood". In German, closed compounds are a common way and highly productive way of creating multi-word expressions. Since most terms are usually longer than one word – Justeson and Katz (1995) find that terms most frequently have a length of two words – closed compounds represent a large proportion of German terms. Furthermore, unithood of closed compound term candidates does not have to be evaluated, since the components are naturally agglutinated and phrase boundaries are evident (in contrast to other multi-word expressions). Instead, compound splitting is required, and the split points are relatively obvious to native speakers, while detecting unithood is not. Another advantage of taking compounds as a basis here is that by addressing complex phrases we can illustrate and exploit the interplay between the termhood tiers of the compounds and their components.

To address the fine-grained termhood prediction of German compound terms, we design a system with three steps: (i) performing compound splitting, (ii) computing features for the compounds and their components, and (iii) applying neural network classifiers to predict the termhood classes. For the neural classifiers, we first adapt the network to German compound words so that it takes information about the components into account. As a second step, we apply a corresponding termhood tier distinction to the compounds' components which further improves the compound class prediction results.

The paper is structured as following: Section 2 describes related work for terminology identification. We illustrate the problem in Section 3 and define the termhood classes. In Section 4, the model for term class prediction is introduced. Section 5 explains the data annotation and evaluation. In Section 6, results are presented and discussed. We conclude in Section 7.

2 Related Work

Approaches for automatic term identification can broadly be classified into four categories: linguistic (Justeson and Katz, 1995; Basili et al., 1997), statistical (Schäfer et al., 2015), hybrid (Frantzi et al., 1998; Maynard and Ananiadou, 1999) and machine learning approaches (Merley da Silva Conrado and Rezende, 2013). Recently, word vector and deep learning approaches (Zadeh and Handschuh, 2014b; Amjadian et al., 2016; Wang et al., 2016) have emerged.

Addressing information about components for evaluation of complex terms (multiword or compound terms) has proven to be effective and was exploited in several termhood measures. The C-value method (Frantzi et al., 1998) is commonly used, which combines linguistic and statistical information and takes nested terms into account for evaluating termhood. The FGM score (Nakagawa and Mori, 2003) computes termhood by taking the geometric mean of the number of distinct left and right neighboring words for each component of a complex term. CSvH (Basili et al., 2001) is a corpora-comparing measure that computes the termhood for a complex term by biasing the termhood score with the general-language frequency of the head. Hätty et al. (2017) combine several termhood measures with a random forest classifier to extract single and multi-word terms and apply the measures recursively to the components. Meanwhile, component information has also been used for the related task of keyphrase extraction. Erbs et al. (2015) split German compounds to enhance individual term frequencies, leading to an improved keyphrase extraction. Zhang et al. (2016) propose a new joint-layer RNN architecture for both classifying keywords and keyphrases. The prediction of the keyword influences the prediction of the keyphrase.

The task of detecting a lay reader's familiarity with certain domain terms, i.e. the terms' difficulty or understandability, is a comparably smaller research area, and a subtask of the more general areas of complex word identification and text readability assessment. It often involves steps of term substitution through simpler synonyms (Kandula et al., 2010) or providing an explanation (Elhadad, 2006).

Notably, Fukushige and Noguchi (2000) adopt an approach for term recognition that involves the concept of term difficulty; they concentrate on the recognition of terms that are both typical to the domain and cannot be easily understood. They motivate this approach by pointing out that these are the terms typically included into a domain-specific glossary.

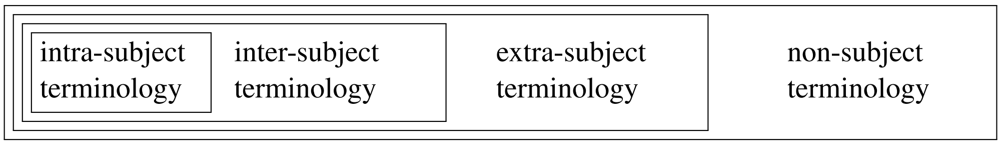
intra-subject terminology
inter-subject terminology
extra-subject terminology
non-subject terminology

Figure 1: Tiers of terminology (Roelcke, 1999), our translation.

3 Termhood Classes and the Role of Closed Compounds

3.1 Defining Termhood Classes

Our goal is to extend the binary notion of termhood and thereby joining the two tasks of automatic term identification and evaluating term difficulty. To achieve this, we propose one task comprising several fine-grained classes of termhood.

The task of automatic term annotation and identification is typically implemented as a binary decision, i.e. recognizing terminology within non-specialized vocabulary. However, from a theoretical point of view, a term candidate can be associated to a domain to different degrees. This is reflected in the definition of *termhood*, and more explicitly in theories describing different kinds of terms (Trimble, 1985; Roelcke, 1999; Tutin, 2007). For example, the model by Roelcke (1999) separates terms into four tiers (Figure 1): *intra-subject terminology* which is specific to the domain, *inter-subject terminology* which is specific to this and other domains, *extra-subject terminology* which does not belong to the domain but is used within it and *non-subject terminology* which is shared across all specific domains. We aimed to create such a tier model with some variation due to the following considerations:

- We want to keep it simple for the annotators, as it is not easy to compare the same term across different domains. We also see no need for that because we want to characterize a domain independently of other domains.

- We consider term difficulty as part of the tier model for a term's strength of association to a domain. It should naturally align to the idea of a gradual increase of term specificity to the domain (as already pointed out in Grabar et al. (2014)): The more difficult or specialized a term is, the more distinctive it is from general language and the more it is associated to a domain. If terms are both general and understandable, it is sometimes hard to distinguish them from general-language words. Thus, the more expert knowledge is needed to understand a term, the stronger it should be associated to a domain.

Considering these points, we introduce the four tiers shown in Table 1: The first tier are the non-terms with no topical relation to the domain, which still may appear in the domain context, such as the word *Deutschland* in *Gericht aus Deutschland* "dish from Germany". The class SIMTERM includes all non-domain terms which are related to the domain to some extent, either by having received a special relevance for the domain (e.g. *Zimmertemparatur* "room temperature" as a temperature measure for dishes) or by semantic relatedness (e.g. *Tiernahrung* "pet food"). The class TERM represents terms which are highly associated to the domain but still understandable. The last class is SPECTERM, whose elements are highly associated to the domain and not understandable for the non-expert reader. With this tier model, we gradually evolve from general to specialized word senses.

3.2 Mutual Impact of Termhood Classes for Compounds and their Components

As already shown in several automatic term identification studies, components of a complex term candidate can be useful for predicting its termhood. We demonstrate in detail how this is the case for our 4-tier model.
In the ideal case, the components define the termhood of the compound:

$$\text{Tomate (TERM) + Püree (TERM)} \rightarrow \text{Tomatenpüree (TERM)}$$
$$\textit{tomato + puree} \rightarrow \textit{tomato puree}$$

Class	Description	Example
NONTERM	not a domain term	*Deutschland* "Germany"
SIMTERM	semantically related to the domain	*Vitaminbedarf* "requirement of vitamins"
TERM	prototypical and understandable term of the domain	*Schweinebraten* "roast pork"
SPECTERM	prototypical and non-understandable term of the domain	*Blausud [blue boiling]* "special kind of boiling fish by adding acid"

Table 1: Termhood classes.

Here, knowledge about the components is sufficient to predict the compound's termhood. So even if there is a lack of information about the compound due to reasons such as low frequency, its termhood can still be predicted on the basis of the components.

However, in other cases knowing the termhood of the components does not necessarily help to directly infer the termhood of the compound. Sometimes the components share the same termhood class, but the compound is a member of a different class.

Example 1:

Mittel (NONTERM) + Alter (NONTERM) → Mittelalter (NONTERM)
mean + age → Middle Ages

Bei (NONTERM) + Fuß (NONTERM) → Beifuß (SPECTERM)
with + foot → mugwort

Example 2:

Mais (TERM) + Anbau (NONTERM) → Maisanbau (SIMTERM)
maize + cultivation → cultivation of maize

Mais (TERM) + Kolben (NONTERM) → Maiskolben (TERM)
maize + cob → maize cob

In the opposite case the compound class is the same but the components' termhood class changes:

Paprika (TERM) + Salat (TERM) → Paprikasalat (TERM)
sweet pepper + salad → sweet pepper salad

Paprika (TERM) + Hälften (NONTERM) → Paprikahälften (TERM)
sweet pepper + halves → halves of sweet pepper

Nevertheless, even in these cases component information is useful because certain termhood classes can be excluded. For example, in the first case, although the component classes are the same, the options for the compound classes are narrowed down to two very different classes. In addition, we expect that a prediction system will learn the interplay between the termhood of compounds and components, and even if the information about a compound is missing, it can be transferred from similarly constructed compounds.

Since especially broad domains like cooking contain many neologisms (e.g. because of new recipe creations: *Parmesanchips* "crisps with Parmesan cheese"), and are anyway very productive, this effect is very advantageous. Many compounds will not be frequent enough to be evaluated for their termhood, and thus components are helpful in evaluating the termhood class.

4 Model for Term Class Prediction

In the following, we describe the model architecture for predicting the previously defined termhood classes (in Figure 2). As mentioned before, we expect information about the compounds' components to be helpful for predicting the termhood classes. For that reason, the compounds need to be split in a first step, to identify the components. Word embeddings and other features will then be computed for both the compound and its components. We test several neural network architectures to predict a compound's termhood class.

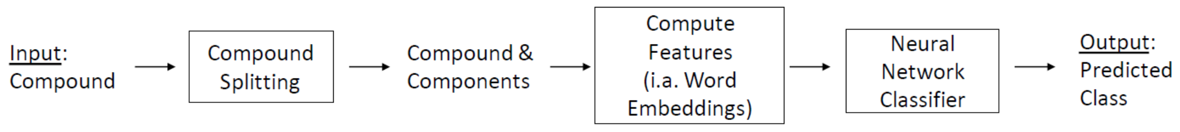

Figure 2: Model architecture.

4.1 Compound Splitting

Compounds are always split into their two main components. We use three splitters:

- *CharSplit* (Tuggener, 2016) is an n-gram-based splitter which is reported to have an accuracy of $\approx$ 95% accuracy for head detection on the GermaNet compound test set.

We further use two morphologically informed splitters.

- *CompoST* (Cap, 2014) is a compound splitter which uses both the morphological resource SMOR (Schmid et al., 2004) and frequency information about components in corpus data for finding the optimal split points. CompoST splits conservatively, leading to a high splitting precision, but many unsplit compounds.

- The *Simple Compound Splitter* (SCS) by Weller-Di Marco (2017) relies on handcrafted morphological rules and frequency information. Weller-Di Marco (2017) compared SCS to an SMOR-based splitter, a variant of CompoST. SCS exhibits higher recall and a higher F1-score.

4.2 Word Embeddings

We compute word embeddings for compounds and components. We use the Word2Vec CBOW model (Mikolov et al., 2013) with a window size of 5 and a minimum frequency of 5 to generate 200-dimensional vectors. The embeddings are pre-trained on Wikipedia, and then trained on the texts from the cooking domain. After compound splitting, 27 components cannot be found, partly because they are infrequent or do not exist as an independent word; but mostly because they are falsely split or the components cannot be lemmatized correctly. To those components we assign a random word vector.

4.3 NN Classifier: Baseline and Hyperparameters

Our basic neural network construction takes one or several concatenated words as input, for which the respective weights in the embedding layer are already set; the weights are initialized with the pre-trained word embedding weights. The final dense layer with a softmax activation predicts the four classes. The architecture is shown in Figure 3.

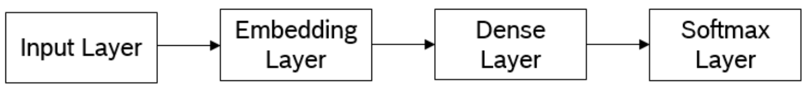

Figure 3: Basic neural network architecture.

The parameters of the neural networks are set to the same values across all network constructions and experiment implementations: the batch size is set to 32, epochs to 50. All dense layers have a dimension of 64. The word embeddings are used to initialize the word weight matrix. The weight matrix is not updated in the training process (since too many parameters would need to be trained). We do not aim for optimizing the model parameters but for comparable architectures, in order to show the effectiveness of the information gained by the input data.

4.4 NNClassifier: Incorporating Compound and Component Information

In the following, we describe the construction of classification models which incorporate both compound and component information. The models build on one another, and we show how additional information about the components leads step by step to a better prediction of the compound termhood class.

Compound and Component Vectors: CONCATVEC In this first model, we stick with the baseline architecture and only modify the input: instead of either relying on the compounds or the component embeddings solely, the network trains on the concatenated embeddings.

Productivity and Compound Frequency: VECPRODFREQ When only taking word embedding information as basis for the prediction, the models receive cumulated information about both the more general domain Wikipedia and the specific cooking domain. However, for the components, additional information about their occurrence in the specific domain is useful. There are two reasons: On the one hand, very specific term parts of the class SPECTERM can be distinguished better from TERM, since these are expected to be less productive and less frequent. For example, the component *blau* 'blue' in *Blausud* (Table 1) refers to a special cooking process and will only occur in this term and the related term *Blaukochen [blue cooking]*. On the other hand, very general or ambiguous components can be tested for their relevance to the domain; for example, components like in *Teigränder* *"rim of pastry"* or *Lorbeerblatt* "bay leaf" are NONTERM. The system can learn that they are nevertheless components which are accepted within a TERM or SPECTERM compound, if they appear in many other compound terms. Thus, we introduce two new features for components:

- Frequency: How frequently does a component appear in other words?
- Productivity: Of how many words the component is part of?

To address this information, we design a shared-layer network architecture as depicted in Figure 4 (but now without the auxiliary output layers). All features – the compound and both the component embeddings, the productivity and frequency of both components – pass through separate layers in the network first, and the information is then concatenated within the network, and compound classes are predicted with a softmax output layer.

Optimization for termhood of components: CONSTOPT As a variant to process the compound and component embedding information more effectively, we finally use a multi-input multi-output shared-layer model, which is depicted in Figure 4. As for the previous model, this one takes five inputs and information is concatenated later on within the network. Additionally, three auxiliary output layers for each the compound and both components are introduced: For the compound, the four described termhood classes are predicted. The auxiliary output layer is a mere regularization mechanism here. For the component, optimization by the auxiliary outputs should sharpen their termhood, since we consider the components' termhood as strongly influential for the termhood of the compound (as explained in detail in section 3). We do not have information about the intrinsic termhood of the component, but we heuristically infer it from the compounds in which a component occurs. As basis, we take all the compounds in the training set and create four classes analogous to the compound classes. Since all combinations would result in too many classes, we make the following distinctions:

- *specific terms*: all components that only appear in SPECTERM compounds; the difference to TERM gets sharpened

- *term*: all components which occur in TERM compounds
- *similar terms*: shared components of SIMTERM and TERM; these SIMTERM components are especially critical, since the other component decides the class (see section 3)
- *other*: the rest of the components

We set the loss weight for the main output to 1, and the weights for the auxiliary outputs to 0.2, which reports the best experiment result.

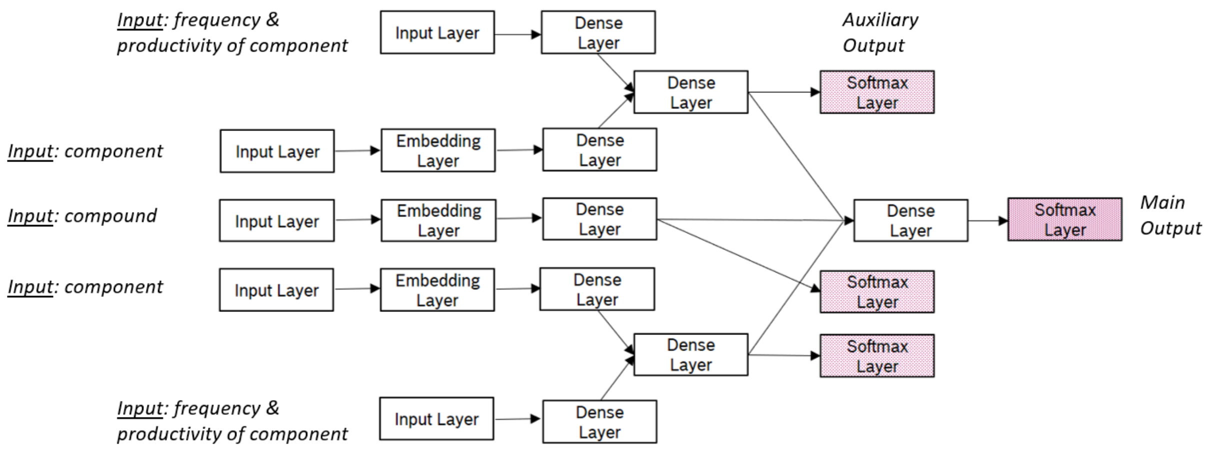

Figure 4: CONSTOPT model.

5 Data, Annotation and Evalaution

We evaluate our approach in the domain of cooking, where 34 cooking recipes are collected randomly from *kochwiki*[1] and *wikibooks*[2]. From these texts, roughly 260 cooking compound terms are identified by three annotators. After briefly reviewing the data, the portion of SPECTERM elements we find is rather low to be used for training (about 10 very specific terms). Therefore, we manually extract compound terms from cooking and cooking-related term lists from Wikipedia. For the classes NONTERM and SIMTERM, we retrieve unannotated compounds of the cooking recipes. We additionally add 15 concepts for the class SIMTERM which have some semantic similarity to the cooking domain (e.g. *Tiernahrung* "pet food", *Küchenzeile* "kitchenette").

For this experiment, the compounds are required to appear sufficiently frequently, such that a word embedding can always be created based on Wikipedia and the cooking domain texts. The reason for this is that we want to show that component information improves the prediction results; thus we want to have an optimal setting for compounds against which the new information can be compared and added to. For the same reason, there is no minimum frequency for the components, to have a realistic setting. We crawl texts from wiki pages (e.g. *kochwiki, wikibooks* and *wikihow*[3]) to attain the minimum frequencies. In total, the collection contains 404 texts from the cooking domain, consisting of roughly 150,000 words. The text collection consists mostly of recipes, but contains descriptions of ingredients or cooking methods as well.

The resulting set of 400 compounds are then prepared for the annotation process: the compound terms are provided with either a definition (from *Wikipedia, kochwiki* or *Wiktionary*[4]) or, if a definition could

[1]`https://www.kochwiki.org/`
[2]`https://de.wikibooks.org/wiki/Kochbuch`
[3]`https://de.wikihow.com/`
[4]`https://de.wiktionary.org/`

not be found, a context sentence from the cooking recipes. Then 5 annotators were asked to decide for one of the four classes NONTERM, SIMTERM, TERM and SPECTERM.

The final classes are selected via majority vote of the annotators. For 46% of the compounds there is a complete agreement (5 out of 5), for 29% there was a 4:1 agreement. For 4 terms, there was no majority voting; these terms are excluded from the compound set, resulting in 396 terms overall.

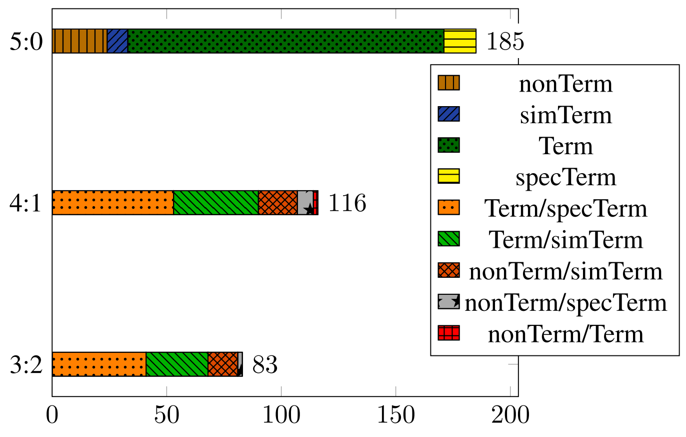

Figure 5: Agreement and disagreement for termhood class annotation.

Figure 5 shows the agreement (5:0) and the disagreement (4:1 and 3:1) for term class annotations. The 5:0-bar shows the number of elements for which the decision is with one consent. These elements are distributed over all classes. For 4:1 and 3:2 disagreements, the number of elements are shown for the two classes in favor of which the decision is made. Annotators are mostly disagreeing between TERM and SPECTERM. This is an expected result, since the knowledge level even of non-expert annotators differs to a certain degree, resulting in a different class selection. Furthermore, nearly all divergences are between neighboring classes (nonTerm/simTerm, Term/simTerm, Term/specTerm) while the divergences between other classes are negligible. This is a good indicator that the class design represents a valid scale of termhood.

After taking the majority vote we get a final distribution of elements per class, as shown in Table 2.

NONTERM	SIMTERM	TERM	SPECTERM
44	43	250	59

Table 2: Number of annotated terms per class.

6 Results and Analysis

6.1 Compound Splitting

Table 3 shows the correct compound splits for the (combinations of) the three splitters. Since CharSplit always splits, we take its splitting performance as baseline. For our specific domain and without special training, CharSplit achieves 91.4% correct splits. There are 25 completely wrong splits (e.g. *Vol|lei* "whole egg" instead of *Voll|ei*), 9 elements have splits on the wrong side (e.g. *Marzipanroh|masse [marzipan raw mass]* "marzipan paste") instead of *Marzipan|rohmasse*).

We further combine splitters to improve the results: Since CompoST splits conservatively and with a high precision, in case of a split we always rely on its output. For combining CompoST and CharSplit, this results in 94.2% correct splits.

Since there are still problems with recognizing the plural -N (*Traube|Nsaft* "grape juice" instead of *Trauben|saft*), we then add the Simple Compound Splitter (SCS), which explicitly models this phenomenon. Since SCS needs a basis of POS, lemma and frequency information, we compute this information on the cooking dataset. Then we combine the three splitters in the way that we again rely first on

CompoST, then apply SCS and finally use CharSplit for the rest of the compounds. This results in the best split score of 95.7%.

Splitters	Wrong splits	Wrong side	% correct splits
CharSplit	25	9	91.4 %
CompoST + CharSplit	14	9	94.2 %
CompoST + SCS + CharSplit	9	8	95.7 %

Table 3: Splitting performances of the three compound splitters.

6.2 Termhood Classification of the Neural Networks

In Table 4, we provide the results for the different models. We apply 5-fold cross-validation and use Precision, Recall and F1-score as evaluation measures. In brackets next to the method name, the overall F1-score is given, weighted and averaged over all classes.

For the baselines, results show that taking only compounds as input provides a better result than only taking component as input. However, note that we provide ideal conditions for compounds, by enforcing a minimum frequency. For the advanced models CONCATVEC, VECPRODFREQ and CONSTOPT, the overall F1-score continuously increases for every improvement we made to the system.

Focusing on the individual classes, NONTERM and TERM achieve a high recognition rate. We attribute this to the fact that they are the two prototypical opposing classes. SEMSIM and SPECTERM reach only decent results at first. When comparing the best model to the best baseline model, the results for the two classes experience a boost. The prediction for SEMSIM gains 17% improvement on the F1-score, SPECTERM even gains 21%.

Method	Non-Term	SimTerm	Term	SpecTerm
Component Baseline [0.69]				
Precision	0.77	0.47	0.78	0.53
Recall	0.50	0.40	0.89	0.32
F1-score	0.6	0.41	0.83	0.37
Compound Baseline [0.72]				
Precision	0.86	0.61	0.79	0.47
Recall	0.77	0.42	0.86	0.37
F1-score	0.80	0.48	0.82	0.40
CONCATVEC [0.75]				
Precision	0.83	0.54	0.82	0.61
Recall	0.68	0.40	0.89	0.54
F1-score	0.74	0.44	0.85	0.57
VECPRODFREQ [0.77]				
Precision	0.85	0.66	0.82	0.62
Recall	0.73	0.53	0.88	0.54
F1-score	0.77	0.58	0.85	0.57
CONSTOPT [0.80]				
Precision	0.88	0.69	0.85	0.64
Recall	0.75	0.63	0.9	0.59
F1-score	0.79	0.65	0.88	0.61

Table 4: Results for baselines and advanced models per class.

Finally, we analyze the predictions of the model to explore reasons for misclassification. One reason behind it could be wrong splits (e.g. *Ei|klar* "egg white", split to *Eik|lar*). However, only 4 out of 17 incorrectly split compounds are predicted incorrectly. Since no embeddings could be computed for wrong components (and a default randomly initialized embedding is taken instead), the model probably learns to rely on other features in this case. The most interesting classes for deeper analysis are SIMTERM and SPECTERM, since the other two classes are predicted well. For SIMTERM, we do not find obvious differences between the correctly classified compounds, and the ones which are predicted as TERM. For SPECTERM, we find at least two kinds of terms that seem to be classified mostly correctly: rather opaque terms (e.g. *Fetthenne [fat hen]* "sedum"), and many terms with one non-understandable, and one understandable component term (e.g. *Tonkabohnen* "tonka beans"). A lot of SPECTERM compounds are classified as TERM. One reason for this might be that a compound includes a very typical TERM component (e.g. *Blindbacken* "blind-baking").

7 Conclusion

We present an approach for combining automatic term identification and evaluation of a term's understandability. We frame it as one task by defining classes representing a fine-grained concept of termhood. As basis, 400 German compounds are selected as candidates for termhood annotation in the domain of cooking. The prediction of these classes is then carried out in a three-step process: applying compound splitting, computing features for compound and components and finally applying a neural network classifier to predict the termhood classes. For the splitting, several existing compound splitters are combined. The features include word embeddings, and productivity and frequency of the components within the specific domain. For the neural network classifier, we design several models to adequately address the information about the compound and its components. A special focus is placed on predicting heuristically approximated classes for the components' termhood, thus improving the prediction of the compound's termhood class as well. The best model achieves a 8% relative improvement on F1-score in comparison to the best baseline model.

The fine-grained notion of termhood introduced here is important for getting a better understanding of the domain. Furthermore, it may be interesting for related topics, such as Keyphrase Extraction and Named Entity Recognition.

As future work, we plan to enlarge the compound basis for both closed compounds and other kinds of multi-word expressions. We especially see potential for the CONSTOPT model, since the basis of complex terms to get evidence for the termhood of components would increase. In addition, we want to include further terminology-specific features into the classifiers, especially corpora-comparing termhood measures. Further we want to explore more deeply the effect of a network trained on complex terms and components features for predicting previously unseen complex terms.

References

Ehsan Amjadian, Diana Inkpen, Tahereh Paribakht, and Farahnaz Faez. 2016. Local-global vectors to improve unigram terminology extraction. In *Proceedings of the 5th International Workshop on Computational Terminology (Computerm)*, pages 2–11, Osaka, Japan.

Mihael Arcan, Marco Turchi, Sara Tonelli, and Paul Buitelaar. 2014. Enhancing statistical machine translation with bilingual terminology in a CAT environment. *Proceedings of the 11th Biennial Conference of the Association for Machine Translation in the Americas (AMTA)*, pages 54–68.

Roberto Basili, Gianluca De Rossi, and Maria Teresa Pazienza. 1997. Inducing terminology for lexical acquisition. In *Proceedings of the 2nd Conference on Empirical Methods in Natural Lanaguge Processing (EMNLP)*, Providence, USA.

Roberto Basili, Maria Teresa Pazienza, Alessandro Moschitti, and Fabio Massimo Zanzotto. 2001. A contrastive approach to term extraction. In *Proceedings of the 4th Terminology and Artificial Intelligence Conference*, pages 119–128.

Gabriel Bernier-Colborne and Patrick Drouin. 2014. Creating a test corpus for term extractors through term annotation. *Terminology*, 20(1):50–73.

Fabienne Cap. 2014. *Morphological Processing of Compounds for Statistical Machine Translation*. Dissertation, Institute for Natural Language Processing (IMS), University of Stuttgart.

Noemie Elhadad. 2006. Comprehending technical texts: Predicting and defining unfamiliar terms. In *AMIA annual symposium proceedings*, volume 2006, pages 239–243. American Medical Informatics Association.

Nicolai Erbs, Pedro Bispo Santos, Torsten Zesch, and Iryna Gurevych. 2015. Counting what counts: Decompounding for keyphrase extraction. In *Proceedings of the ACL 2015 Workshop on Novel Computational Approaches to Keyphrase Extraction*, pages 10 – 17, Beijing, China.

Katerina T. Frantzi, Sophia Ananiadou, and Jun-ichi Tsujii. 1998. The c-value/nc-value method of automatic recognition for multi-word terms. In *Proceedings of the 2nd European Conference on Research and Advanced Technology for Digital Libraries*, pages 585–604, London, UK.

Yoshio Fukushige and Naohiko Noguchi. 2000. Statistical and linguistic approaches to automatic term recognition: NTCIR experiments at matsushita. *Terminology*, 6(2):257–286.

Natalia Grabar and Thierry Hamon. 2014. Unsupervised method for the acquisition of general language paraphrases for medical compounds. *Proceedings of the 4th International Workshop on Computational Terminology (Computerm)*, pages 94–103.

Natalia Grabar, Thierry Hamon, and Dany Amiot. 2014. Automatic diagnosis of understanding of medical words. In *Proceedings of the 3rd Workshop on Predicting and Improving Text Readability for Target Reader Populations (PITR)*, pages 11–20.

Anna Hätty, Michael Dorna, and Sabine Schulte im Walde. 2017. Evaluating the reliability and interaction of recursively used feature classes for terminology extraction. In *Proceedings of the 15th Conference of the European Chapter of the Association for Computational Linguistics, Student Research Workshop*, pages 113–121, Valencia, Spain.

John S. Justeson and Slava M. Katz. 1995. Technical terminology: Some linguistic properties and an algorithm for identification in text. *Natural Language Engineering*, 1(1):9–27.

Kyo Kagueura and Bin Umino. 1996. Methods of automatic term recognition: A review. *Terminology*, 3(2):259–289.

Sasikiran Kandula, Dorothy Curtis, and Qing Zeng-Treitler. 2010. A semantic and syntactic text simplification tool for health content. In *AMIA annual symposium proceedings*, volume 2010, pages 366–370. American Medical Informatics Association.

Diana Maynard and Sophia Ananiadou. 1999. Identifying contextual information for multi-word term extraction. In *Proceedings of 5th International Congress on Terminology and Knowledge Engineering*, pages 212–221.

Thiago Alexandre Salgueiro Pardo Merley da Silva Conrado and Solange Oliveira Rezende. 2013. A machine learning approach to automatic term extraction using a rich feature set. In *Proceedings of the North American Chapter of the Association for Computational Linguistics - Human Language Technologies*, pages 16–23, Atlanta, Georgia, USA.

Tomas Mikolov, Ilya Sutskever, Kai Chen, Greg Corrado, and Jeffrey Dean. 2013. Distributed representations of words and phrases and their compositionality. In *Advances in Neural Information Processing Systems (NIPS)*, pages 3111–3119.

Hiroshi Nakagawa and Tatsunori Mori. 2003. Automatic term recognition based on statistics of compound nouns and their components. *Terminology*, 9(2):201–219.

Martin Riedl and Chris Biemann. 2015. A single word is not enough: Ranking multiword expressions using distributional semantics. In *Proceedings of the 2015 Conference on Empirical Methods in Natural Language Processing (EMNLP)*, Lisboa, Portugal.

Thorsten Roelcke. 1999. *Fachsprachen*. Grundlagen der Germanistik. Erich Schmidt Verlag.

Johannes Schäfer, Ina Rösiger, Ulrich Heid, and Michael Dorna. 2015. Evaluating noise reduction strategies for terminology extraction. In *Proceedings of Terminology and Artifical Intelligence*, pages 44–49, Granada, Spain.

Helmut Schmid, Arne Fitschen, and Ulrich Heid. 2004. A German computational morphology covering derivation, composition, and inflection. In *Proceedings of the 4th International Conference on Language Resources and Evaluation (LREC)*, pages 1263–1266, Lisbon, Portugal.

Louis Trimble. 1985. *English for Science and Technology: A Discourse Approach.* Cambridge Univ. Press.

Don Tuggener. 2016. *Incremental Coreference Resolution for German.* Dissertation, Faculty of Arts, University of Zurich.

Agnes Tutin. 2007. Traitement sémantique par analyse distributionelle des noms transdisciplinaires des écrits scientifiques. *In Actes de TALN*, pages 283–292.

Juan Antonio Lossio Ventura, Clement Jonquet, Mathieu Roche, and Maguelonne Teisseire. 2014. Yet another ranking function for automatic multiword term extraction. In *Advances in Natural Language Processing - 9th International Conference on NLP*, pages 52–64, Warsaw, Poland.

Rui Wang, Wei Liu, and Chris McDonald. 2016. Featureless domain-specific term extraction with minimal labelled data. In *Proceedings of the Australasian Language Technology Association Workshop 2016*, pages 103–112, Melbourne, Australia.

Marion Weller-Di Marco. 2017. Simple compound splitting for German. In *Proceedings of the 13th Workshop on Multiword Expressions (MWE)*, pages 161–166, Valencia, Spain.

Behrang Zadeh and Siegfried Handschuh. 2014a. The acl rd-tec: A dataset for benchmarking terminology extraction and classification in computational linguistics. In *Proceedings of the 4th International Workshop on Computational Terminology (Computerm)*, pages 52–63, Dublin, Ireland.

Behrang Zadeh and Siegfried Handschuh. 2014b. Evaluation of technology term recognition with random indexing. In *Proceedings of the 9th International Conference on Language Resources and Evaluation (LREC)*, pages 4027–4032, Reykjavik, Iceland.

Qing Zeng-Treitler, Sergey Goryachev, Tony Tse, Alla Keselman, and Aziz Boxwala. 2008. Estimating consumer familiarity with health terminology: A context-based approach. *Journal of the American Medical Informatics Association*, 15(3):349–356.

Qi Zhang, Yang Wang, Yeyun Gong, and Xuanjing Huang. 2016. Keyphrase extraction using deep recurrent neural networks on twitter. In *Proceedings of the 2016 Conference on Empirical Methods in Natural Language Processing (EMNLP)*, pages 836–845, Austin, TX, USA.