Research Group Social Computing
Department of Informatics
Technical University of Munich

Weakly Supervised Aspect Extraction Using a Student-Teacher Co-Training Approach

Report for NLP Lab Course

Jingpei Wu ✉, **Ke Xin Chen** ✉, and **Kevin George** ✉

Department of Informatics, Technical University of Munich

✉ jingpei.wu@tum.de
✉ ge59kuv@mytum.de
✉ kevin.george@tum.de

July 31, 2021

Abstract — Aspect detection is a salient task of sentiment analysis and opinion mining. Different aspects of a particular entity (e.g., price, quality, safety), can be identified based on text reviews/comments of the entity. While the supervised approaches for aspect classification require many fine-grained labels, unsupervised models often fail to accurately extract the aspects. Weakly supervised models like the one presented in this project, require only a small set of 'seed words' per aspect. We explore the student-teacher co-training approach, which leverages just a few seed words in a bag-of-words teacher classifier which subsequently trains a second student (neural network) model. First, we extract seed words from the organic dataset using three different approaches. Then, we experiment with various word embeddings that are pre-trained and finetune them to our organic dataset. Next, we use the student-teacher classifiers to predict the aspects based on Iterative Seed Word Distillation and show that the iterative co-training can predict aspects based only on input seed words. Finally, we illustrate the results from experimenting with different seed word extraction methods and using different word embeddings for aspect extraction.

1 Introduction

With the rapid growth of the internet and notably social media, the number of people expressing their opinions on the Web has increased. Based on these increasing treads and the potential for numerous information access applications, individuals and organizations are analyzing these user-generated reviews for their decision making [7].

Aspect detection is an important task of sentiment analysis and opinion mining which aims to identify the aspect categories in a given text. (e.g., "price" and "image quality" in TV product reviews) [12]. The majority of work in detection learning focuses on supervised and unsupervised approaches. However, supervised approaches heavily rely on hand-labeled training data, which are expensive and time-consuming to create [10]. Unsupervised approaches will not guarantee the same level of accuracy in capturing aspects of interest. In this work, we consider using a weak supervision approach, as this method minimizes labeling effort and has better behavior than unsupervised approaches. We refer to a student-teacher co-training weak supervision approach that has been introduced by Karamanolakis et al. [5]. This method trains aspect classifiers by stating only a few seed words per aspect where the seed words are weakly positive indicators for the aspects of interest [5].

In this project, we investigate the findings of the paper [5]. We also investigate the effects of using different methods for deriving seed words and using more recent word embedding techniques for training aspect classifiers. In addition to it, we perform these tasks on the English organic dataset, which contains comments from social media about organic food products. Eventually, we apply and evaluate the model on the German organic dataset.

First, we found an unofficial implementation for the co-training paper [5]. We reviewed this implementation in detail and corrected some errors based on our understanding of the algorithms. We used this implementation as the basis for our implementation. Next, we derived the seed words using three different methods, namely word vectors, NMF, and Clarity Scoring Function. Finally, we applied the approach with various fine-tuned word embedding methods, including BERT, Word2Vec, and GloVe.

The rest of this paper is organized as follows: In Section 2, we describe our methods on pre-trained models, seed words, and student-teacher approach. In Section 3, we present our experimental setup. In Section 4, we demonstrate our results. Finally, in Section 5, we conclude and suggest future improvements.

2 Methods

We now formulate our problem and then describe the related methodologies that we use in the project.

2.1 Problem Definition

The corpus of text reviews from a domain (e.g., televisions, organic products) is split into segments (e.g., sentences, clauses). We now consider K pre-defined aspects of interest $(1, \dots, K)$, including the "General" aspect, which we assume is the 0-th aspect for simplicity. Different segments of the same review may be associated with different aspects but, ground-truth aspect labels are not available for training. Instead, a small number of seed words G_k are provided for each aspect $k \in K$. Our goal is to use the corpus of training reviews and the available seed words $G = (G_1, \dots, G_K)$ to train a classifier, which, given an unseen test segment s, predicts K aspect probabilities $p = (p_1, \dots, p_K)$.

2.2 Pre-trained Models

For this classification problem, we choose three popular pre-trained models, i.e. Word2Vec [8], GloVe [9] and BERT [3], for comparison. They are trained in different ways and here we introduce the basic ideas and the concrete implementation details for each of them.

2.2.1 Word2Vec

There are two types of architectures in Word2Vec, skip-gram and continuous bag-of-words (CBOW). In skip-gram, given surrounding words in a window, the prediction probability of the current word $p(w|c)$ is to be maximized. Here w is the current word, c is the context of w, and words close to w within a definite range are called window size. CBOW is the reverse. It predicts context words given the current word, which means probability $p(c|w)$ is to be maximized.

In our approach, we use the data structure from library gensim[1]. The pre-trained model[2] contains 300-dimensional vectors for 3 billion words and phrases trained on the Google News Dataset. For fine-tuning, we extract approximately 26000-word vectors using the built-in API provided by gensim, and then train for 100 epochs. In terms of hyperparameter setting, the default values are chosen, i.e. window size is 5 and the minimum word frequency count is also 5. We use the default value since we are not focused on fine-tuning the embeddings rather on the student-teacher iterative co-training implementation. Group 1.2[3] in the same practical course has shown that the default setting can already yield good performance with hardly any improvement during hyperparameter tuning.

2.2.2 GloVe

GloVe (Global Vectors for Word Representation) is an unsupervised learning algorithm for obtaining vector representations for words. GloVe is essentially a log-bilinear model with a weighted least-squares objective. The main intuition underlying the model is the simple observation that ratios of word-word co-occurrence probabilities have the potential for encoding some form of meaning. The model is trained on the non-zero entries of a global word-word co-occurrence matrix, which tabulates how frequently words co-occur with one another in a given corpus. Populating this matrix requires a single pass through the entire corpus to collect the statistics. For large corpora, this pass can be computationally expensive, but it is a one-time up-front cost. The resulting representations showcase interesting linear substructures of the word vector space.

The training objective of GloVe is to learn word vectors such that their dot product equals the logarithm of the words' probability of co-occurrence. Since the logarithm of a ratio equals the difference of logarithms, this objective associate (the logarithm of) ratios of co-occurrence probabilities with vector differences in the word vector space.

For this project, we used pre-trained word vectors made available by Stanford NLP [4]. In particular, we used the 300-dimensional vectors of GloVe from Wikipedia 2014 + Gigaword 5 which contains 6B tokens and 400K vocab. Finetuning the model to the organic dataset was attempted using the Mittens library [4]. The authors of Mittens mention that the implementation is only suitable for modest vocabularies (up to 20k tokens) since the co-occurrence matrix must be held in memory. Our experiments hit memory limits at 12k tokens while the vocabulary was several times larger. Therefore fine-tuning the entire vocabulary proved difficult.

[1]https://radimrehurek.com/gensim_3.8.3/index.html
[2]https://code.google.com/archive/p/word2vec/

[3]https://gitlab.lrz.de/social-computing-research-group/nlp-lab-course/nlp-lab-course-ss2021/opinion-mining/group-1.2
[4]https://nlp.stanford.edu/projects/glove/

2.2.3 BERT

BERT (Bidirectional Encoder Representations from Transformers) is a fine-tuning-based language representation model applying the bidirectional pre-training with only unlabeled text corpus. It is powerful and easy to fine-tune compared to the previous unidirectional language models. In the following content, we describe the details of BERT, including its way of text presentation, pre-train steps, architecture, and fine-tuning detail.

BERT has its own way of text representation, where it receives pairs of sentences as input. It uses a special token to distinguish the two sentences in the received pair input, a [CLS] token is to indicate the beginning of the first sentence and an [SEP] token to indicate the end of each sentence. The pre-training procedure follows the existing literature on language model pre-training. The author uses the BooksCorpus (800M words) and English Wikipedia (2,500M words) as pre-training corpus. There are two tasks to pre-train BERT. First, it uses a "masked language model" (MLM) pre-training objective, which reduces unidirectionality constraints. For example, OpenAI GPT, uses a left-to-right architecture, as each token can only access previous tokens, whereas BERT can access both left and right context. Additionally, BERT uses a "next sentence prediction" (NSP) task, which intends to predict if the second sentence in the received pair is the actual next sentence that follows the first sentence in the original text. The architecture of BERT is a multi-layer bidirectional Transformer encoder. The BERT base model includes 12 layers, a hidden size of 768, 12 self-attention heads, and 110M total parameters. The fine-tuning of BERT is straightforward as most model hyperparameters are the same as in pre-training. The author has suggested some values for each tunable hyperparameter to use across all types of tasks: Batch size: 16, 32; Learning rate (Adam): 5e-5, 3e-5, 2e-5; Number of epochs: 2, 3, 4; Dropout probability: always kept at 0.1. We have followed most of the suggestions when implementing our own tasks.

2.3 Seed Words

Seed words play a crucial role in our approach since they represent each aspect, provide pseudo labels for the classifier, and thus directly affect the performance of the whole model. They should cover as many sentences that are in the corresponding aspect as possible. To derive high-quality seed words effectively, we try three different approaches, illustrated as follows.

2.3.1 Word Vectors

Word vectors show the ability to capture semantic meanings of words after training with a large corpus and some relationships between words are encoded in the embedding space. The most common example is $v(\text{king}) - v(\text{queen}) \approx v(\text{man}) - v(\text{woman})$, with $v(\text{king})$ representing the word vector of king. Words that share similar contexts are more likely to talk about the same aspects. So words close to each other in the embedding space are expected to belong to the same aspect. More specifically, we run the k-means algorithm on the fine-tuned Word2Vec vectors and choose n words closest to each centroid as the seed words for this aspect. For the best number of aspects k, we use the elbow method, Akaike information criterion (AIC)[5] and Bayesian information criterion (BIC)[6] as reference (both lower is better).

2.3.2 NMF

NMF [6] (Non-negative matrix factorization) is a dimension reduction and factor analysis method which is similar to Principal Component Analysis (PCA). It has been widely used as an unsupervised document clustering and topic modeling method in NLP. Given the original matrix A, the goal is to obtain two matrices W and H with a desired lower dimension such that:

$$A \approx WH \tag{1}$$

In our work, A is the document-word matrix which represents the input that contains words from our documents. W is the basis vector that represents the topics found from the input document. H is the coefficient matrix that represents the weights for the topics. The matrices W and H are found using the Frobenius norm (a method to measure the distance between two given matrices) as an optimization problem. Below is the most commonly used formulation for NMF based on the Frobenius norm:

$$\min_{W \geq 0, H \geq 0} f(W, H) = \|A - WH\|_F^2 \tag{2}$$

By applying NMF with our organic dataset, we can see that some of the seed words defined in each aspect are related but not as clear as the next method: Clarity Scoring Function.

[5]https://en.wikipedia.org/wiki/Akaike_information_criterion
[6]https://en.wikipedia.org/wiki/Bayesian_information_criterion

2.3.3 Clarity Scoring Function

Clarity measures how much more likely it is to observe word w in the subset of segments that discuss aspect a, compared to the corpus as a whole.

$$\text{score}_a(w) = t_a(w) \log_2 \frac{t_a(w)}{t(w)} \tag{3}$$

where $t_a(w)$ and $t(w)$ are the L1-normalized tf-idf scores of w in the segments annotated with aspect a and in all annotated segments, respectively. Higher scores indicate higher term importance and truncating the ranked list of terms gives a fixed set of seed words, as well as their seed weights by normalizing the scores to add up to one.

Finally, we chose the Clarity Scoring Function as our primary method. Both [1] and [5] apply Clarity Scoring Function to the validation set to derive seed words, leaving the test set for evaluation. In our case, for simplicity, we directly use the annotated dataset[7] for choosing seed words, which seems against the idea of weak supervision, which is reducing the need for label efforts. However, in this way, we can have quantitative results to compare with the reported result in the paper [5] and debug our code easily.

2.4 Student-Teacher Approach

2.4.1 Teacher

The teacher is a bag of words classifier that leverages the seed words G for each of the K aspects. For each segment s in the corpus, the teacher predicts aspect probabilities for each aspect using the seed words G_k. The teacher uses the seed words to produce an intuitive prediction q_i which are proportional to the counts of the seed words under G_k:

$$q_i^k = \frac{\exp\left(\sum_{j=1}^{D} \mathbb{1}\{j \in G_k\} \cdot c_i^j\right)}{\sum_{k'} \exp\left(\sum_{j=1}^{D} \mathbb{1}\{j \in G_{k'}\} \cdot c_i^j\right)} \tag{4}$$

where c_i^j encodes the number of times the j-th seed word occurs in s_i. If no seed words appear, the teacher sets the prediction to the *General* aspect. This classifier ignores all the non-seed words to make the preciction. Here, the teacher considers all the seed words for an aspect with equal value.

[7]https://gitlab.lrz.de/social-rom/datasets/organic-dataset/annotated-dataset/-/tree/master/annotated_3rd_round

While the teacher only uses seed words to make predictions, the non-seed words are likely to have an influence. This is where the student network comes in.

2.4.2 Student

The student model is an embedding-based neural network where each segment s_i is first embedded via one of the previously mentioned embedding methods and then classified to the K aspects. The student does not train using any ground truth labels, instead, it attempts to optimize the distillation objective. The distillation objective refers to the cross-entropy between the teacher's predictions q_i and the student's predictions p_i:

$$H(q_i, p_i) = -\sum_k q_i^k \log p_i^k \tag{5}$$

Unlike the teacher, the student network uses all the words in a segment s_i and additionally learns to associate non-seed words to the K aspects. As a result, the student can generalize better and predict aspects for segments that do not contain any seed words.

2.4.3 Seed Word Quality

As mentioned in section 2.4.1, the teacher tends to consider each seed word with an equal weightage. However, this is problematic because not all words are equally good to predict an aspect. Hence, seed words are now considered as "noisy annotators" and a quality parameter is introduced.

The predictive quality of the j-th seed word is introduced as a weight vector $z_j = (z_j^1, ..., z_j^k)$ where z_j^k measures the strength of the association with the k-th aspect. Hence, Equation [4] is now replaced by:

$$q_i^k = \frac{\exp\left(\sum_{j=1}^{D} \mathbb{1}\{j \in G_k\} \cdot \hat{z}_j^k \cdot c_i^j\right)}{\sum_{k'} \exp\left(\sum_{j=1}^{D} \mathbb{1}\{j \in G_{k'}\} \cdot \hat{z}_j^{k'} \cdot c_i^j\right)} \tag{6}$$

where $\hat{z}_j$ is the current estimate of z_j.

Further, we take the student's predictions for a segment s_i to be $t_i = argmax_k\ p_i^k$ and use it to get the current estimate $\hat{z}_j$.

$$\hat{z}_j^k = \frac{\sum_{i=1}^{N} \mathbb{1}\{c_i^j > 0\}\mathbb{1}\{t_i = k\}}{\sum_{k'} \sum_{i=1}^{N} \mathbb{1}\{c_i^j > 0\}\mathbb{1}\{t_i = k'\}} \tag{7}$$

Now the quality of the j-th seed word is estimated according to the student-teacher agreement on segments where the seed word appears.

2.4.4 Iterative Seed Word Distillation (ISWD)

Co-training [2] is a classic multi-view learning method for semi-supervised learning. In co-training, classifiers over different feature spaces are encouraged to agree in their predictions on a large pool of unlabeled examples. Taking the previously presented ideas in Section 2.4 and the idea of co-training, Iterative Seed Word Distillation (ISWD) is introduced. The main steps in ISWD are:

1. Apply the teacher on unlabeled training segments to get predictions (without considering seed word qualities).

2. Train the student using the teacher's predictions (optimize the cross-entropy between the teacher's (soft) predictions and the student's predictions).

3. Apply the student on the training data to get predictions.

4. Update the seed word quality parameters using the student's predictions.

5. Iterate Steps 1-4 until convergence.

3 Experiments

3.1 Dataset

In our project we use two datasets, OPOSUM[1] and Organic dataset[8] (including its annotated version[7]).

OPOSUM is used in our main reference paper[5], containing Amazon product reviews from six domains (bags_and_cases, Bluetooth, boots, keyboards, tv, vacuums), each of which has nine aspects. The details are available at [1]. We use it mainly for comparison and verifying the code's correctness.

Organic dataset, which collects articles and comments from different social media websites about organic food products, is our primary target. The annotated version is labeled with entities and attributes. Attributes are closer to the concept of aspects than entities and for a stable training process, it is better to have more samples in each aspect. Hence, we use the coarse attributes as our labels, with 6 classes instead of 14 classes from the fine attributes. The final aspects can be seen in Table 1.

[8]https://gitlab.lrz.de/social-rom/organic-dataset/curated-source-dataset/-/tree/master/english

3.2 Code

Our code is based on different repositories and we want to acknowledge it here. The oposum dataset is processed and saved with hdf5 files according to the repository[9], from where we infer the structure of the data and how to extract useful data.

For the student-teacher approach, we found an unofficial repository[10] based on the original paper [5] when we started to implement the approach. Our implemented results for domain bags_and_cases are not the same as stated in their readme file. After reading the code in detail, several inconsistencies between the code and the approach description in paper [5] are found. One of them is that the student-teacher co-training iteration is executed inside each mini-batch instead of looping over the whole corpus. In other words, student parameters and seed words weights are updated iteratively using batch data as illustrated in 2.4.4 and the seed word weights are reset to equal at the beginning of each mini-batch. We are doubtful if the seed word weights are representative and stable enough since the batch size is only 64. So we try to accumulate seed words weights along different mini-batches and not reset them in each mini-batch. Another alternative is that we firstly train the student over the corpus to make it as close to the teacher as possible, and then calculate the seed words weights with the updated student once for the whole corpus and then update the teacher. Both of these trials didn't give promising improvement, yet we changed the student-teacher iterative co-training procedure within the corpus since it seems closer to the experiments stated in the paper [5]. Another inconsistency is, for those sentences without any seed words, it is assumed that the seed words appear once in the *General* aspect. However, the probability output for aspects after softmax is not a one-hot vector, which is slightly different from assigning the sentence *General* aspect. The results showed that this minor change helped to improve the performance.

At the last stage of the project, i.e. the end of June, we found that the official code[11] for [5] was published. As we were already familiar with our temporal code, we decided not to switch to this pipeline but to go through this official code to ensure that we did not miss any critical detail. The results in [5] can be reproduced following the default setting in the repository, but the iterative co-training is not executed at all, which seems contradictory to the paper. Code is present for calcu-

[9]https://github.com/stangelid/oposum
[10]https://github.com/aqweteddy/LeverageJustAFewKeywords
[11]https://github.com/gkaramanolakis/ISWD

lating, storing, and loading seed words weights, but the data are just loaded once at the beginning without seed words weights and the weights are not updated during the training process. It means that the reported results come out from a simple classifier on top of the vector representation of sentences, even without a complex mechanism like attention. However, we add techniques like weight decay, dropout, options for frozen embedding, etc. to our code.

3.3 Experimental Procedure

OPOSUM dataset use tokenization, stemming and removing stopwords as text pre-processing techniques. All required processed data can be extracted from their repository. We apply our code using fine-tuned Word2Vec model to these data and compare the average f1 score with the result in the paper [5], with the same hyperparameter setting.

In the provided organic dataset, comments are cleaned up but the articles are in their raw status. We just need comments for aspect extraction. But in order to have more robust fine-tuned word embeddings we want more data and hence articles are used. We apply the same clean-up techniques as comments to articles. Then we use the same pre-processing pipeline to articles and comments. Based on the results from [11], we also remove words that only appear once.

The detail procedure of deriving seed words and fine-tuning word embeddings are already stated in 2.3 and 2.2. For a fair comparison, we also use 30 seed words for each aspect.

In terms of the final classification problem, we apply each word embedding models (fine-tuned Word2Vec, GloVe and base uncased BERT) and report the average best test performance over 5 different runs. We execute hyperparameter tuning with one run in each configuration for quick intuition. The chosen hyperparameters are learning rate, weight decay coefficient and dropout. The result is shown in Figure 1 and we chose the hyperparameters that produce better performance more frequently. Additionally, we also run experiments on non-frozen embeddings or non-fine-tuned embeddings for comparison.

As evaluation metrics, we use both the micro- and macro-averaged F1 score. Precision and recall are also used to analyze the outcome. We also look into the agreement ratio, which is the percentage of samples with the same result from both teacher and student model.

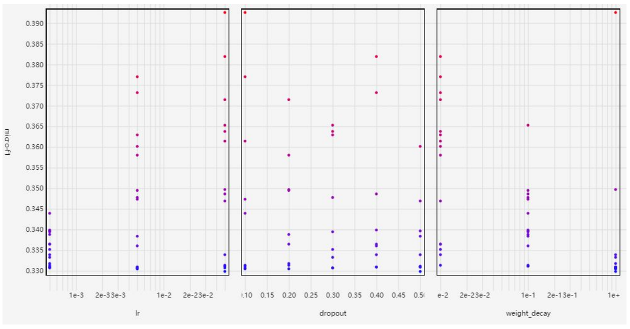


Figure 1 Scatter plot for hyperparameter tuning with Word2Vec

4 Results

In this section we show and analyze the results from seed words and student-teacher approach.

4.1 Seed Words

The idea of running clustering algorithm on embedding space did not give promising result. With ten clusters, only one cluster showed meaningful aspect, which was related to 'chemical items' while all the other clusters were quite noisy. To choose the best number of clusters, both elbow method and AIC/BIC did not provide useful information. The figure 2 showed that small k is best, which did not make sense in our case.

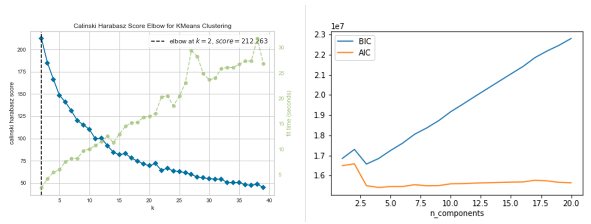


Figure 2 Choosing best k for k-means algorithm: left is elbow method, right is AIC/BIC

By using NMF to derive seed words, only a few words in each aspect are actually related and are not as clear as the results from the Clarity Scoring Function.

Finally the result from Clarity Scoring Function is used for the experiments and some example seed words are shown in Table [1].

4.2 Word2Vec

Comparison on OPOSUM In order to make sure that our code is right, the appropriate way is to compare with the reported result in paper [5]. The result is shown in table 2. Our implementation is not

Aspect	Seed Words
General	farming, store, organic, India, definition
Price	price, expensive, cost, pay, money
Experienced Quality	taste, nutrient, better, quality, superior
Safety and Healthiness	pesticide, chemical, health, fertilizer, toxic
Trustworthy Sources	label, certified, certification, market, illegal
Environment	environment, soil, impact, sustainable, resource

Table 1 Top 5 seed words for each aspect.

competitive in most of the domains, while performs way better in domain boots. As we mentioned in section 3.2, we have tried different variants but gained no improvement and we did not get much information from the official repository. Besides, the word vectors in official code are pre-defined, which means we do not know how they fine-tune them.

Experiments on organic dataset The hyperparameter configuration is the same as GloVe. The best test result appear in the second iteration of co-training, with average micro f1 score 0.363 and 80% agreement ratio between teacher and student. Here we show the confusion matrix from the prediction in Figure 3. If we look at the diagonal, many aspects show acceptable percentage of true positives and false positives, except that there is no *Trustworthy Sources* predictions and most of the textitExperienced Quality aspects are wrongly predicted. There could be some improvement from this point of view, which is left for future work.

4.3 GloVe

Glove was run with the following parameter configurations:

- Batch size: 16
- Learning rate (Adam): 5e-4
- Number of epochs: 4
- Frozen Embeddings
- Dropout: 0.4
- Weight Decay: 0.01

Batch sizes were varied to 8 to 1024. All the scores seem to be slightly decreasing with increasing batch size. The micro f1 score is seen to be increasing from 0.29 to 0.33 over 4 iterations. The agreement ratio is constant around 0.71. Meanwhile, the macro f-1 score is also increasing from 0.22 to 0.23. The class distribution of the predictions shows some faithfulness to the Ground Truth distribution although the two vary by several percentages. A strong difference in *Experienced Quality* and *Trustworthy Sources* is observed which may be attributed to the similarity in the seed words of the respective classes.

4.4 BERT

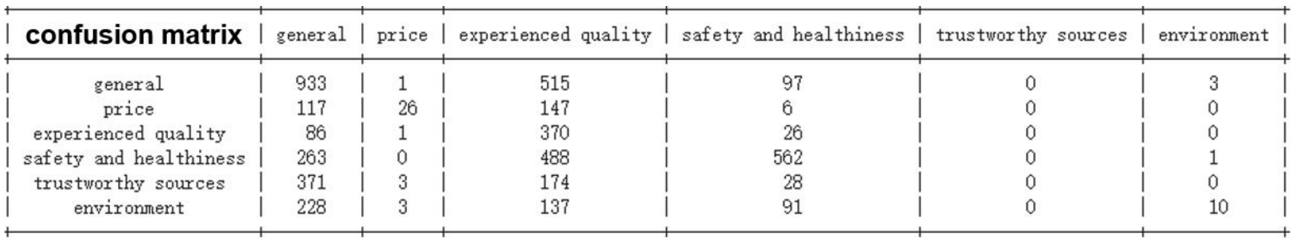
After testing with different configurations, the following parameter configuration achieves the best validation performance for BERT:

- Batch size: 256
- Learning rate (Adam): 5e-5
- Number of epochs: 4
- No embedding freezing
- Run per mini-batch instead of looping over the whole corpus.

As an overview of the results shown in Figure 4, the micro f1 score, and macro f1 score are slightly increasing per epoch. The average micro f1 score increases from 0.34 to 0.36, the average macro f1 score increases from 0.12 to 0.17, and the average agreement ratio increases from 0.89 to 0.91.

The class distribution shown in Figure 6 is very unbalanced compared to the ground truth distribution. The *General* class takes 92% over the test dataset, whereas the *Environment* class only takes 0.34% over the test dataset. On analyzing the root cause of these unbalanced predictions, we understand that using only 30 seed words per aspect is still too weak for labeling the sentence. We suggest using a larger number of seed words in future works.

To perform qualitative analysis, we chose some outcomes to see how well the classifiers derive the results. We look into the *Price* class, as it is easy to evaluate with common 'human sense'. The following results are predicted as *Price* class for both experiments that we ran. From the outcomes, we see some positive samples. For example, *"Anyone can be bought"*, *"Money makes rationalizing evil palatable to most people"* and *"Usually the Organic farms are regulated more and have fees they have to pay to our Gov. to be Organic. That's more than likely why they cost more"*. These samples are clearly related to the *Price* aspect. We also



confusion matrix	general	price	experienced quality	safety and healthiness	trustworthy sources	environment
general	933	1	515	97	0	3
price	117	26	147	6	0	0
experienced quality	86	1	370	26	0	0
safety and healthiness	263	0	488	562	0	1
trustworthy sources	371	3	174	28	0	0
environment	228	3	137	91	0	10

Figure 3 Confusion matrix from the prediction of Word2Vec on organic dataset

	bags	keyboards	boots	headsets	TVs	Vacuums
paper [5]	**59.3**	57.0	48.3	**66.8**	**64.0**	57.0
our experiment	47.6	53.6	**52.6**	54.0	54.2	54.6

Table 2 Micro-f1 score comparison between reported result in [5] and our implementation using Word2Vec embedding. Bold implies far better than the other implementation.

see some negative samples. For example, *"there is no such thing as all-natural unless brands/companies are transitioning to organic!"* and *"why should not they? are you opposed to governments taking action to protect the environment and public health?"*. We classified them as negative as we do not see a clear relationship with the *Price* aspect. In conclusion, the model made an appreciable effort with respect to the results given that it works based on only 30 seed words per aspect. However, it does not always predict the correct aspect.

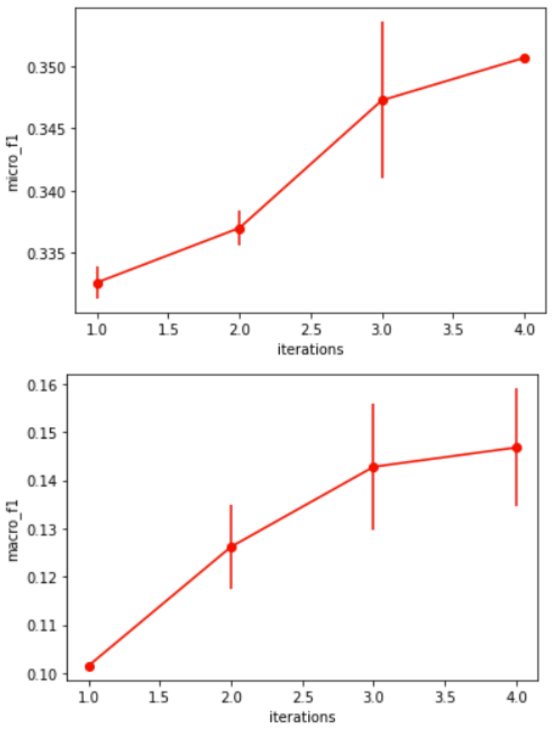


Figure 4 Micro/Macro f1 score for BERT

4.5 German Organic Dataset

We used *"de_core_news_md"* package from Spacy, and german stopwords package from nltk for preprocessing our German dataset (tokenization, stemming, and removing stopwords).

As we do not have the annotated German dataset, we cannot perform the Clarity Scoring Function to derive the seed words as we did for our English dataset. We are also not able to test the model with a labeled dataset. As a result, we use NMF instead of k-means algorithms, since NMF performs better in our English organic dataset. Table 3 shows the top 5 seed words per each of the 6 aspects by applying NMF, but we can see that only some of the words are related to the same topic and it is difficult to define the name of the aspect.

For pre-training and testing, we plan to use "bert-base-german-cased" and then inspect some results to see how the model performs. Unfortunately, we have not finished working with the German dataset. This process only suggests future work.

Aspect	Seed Words
Topic 0	Zitat, eier, skandal, dioxin, kontrollen
Topic 1	Fleisch, tiere, gemüse, eier, woche
Topic 2	Bauern, milch, bauer, landwirtschaft, subventionen
Topic 3	Menschen, tiere, natur, mensch, welt
Topic 4	Bio, produkte, eier, gemüse, bioprodukte
Topic 5	Lebensmittel, geld, produkte, leute, qualität

Table 3 Top 5 seed words per aspect for German dataset

	Micro F1	Macro F1	Agreement Ratio
Word2Vec	0.363	0.210	0.8
GloVe	0.333	0.232	0.709
BERT	0.359	0.167	0.906

Table 4 Measure scores for each word embedding

4.6 Result Comparison

We compare the three word embedding models Word2Vec, Glove and BERT from different views and aspects.

4.6.1 Measure Scores Comparison

As seen in Table 4, we do not see a large difference on micro F1 score as all the three word embedding methods get around 0.35. However, we can see that there is a difference when comparing macro f1 score and agreement ratio. It is interesting to see that even though GloVe has the lowest micro F1 score, it has the highest macro F1 score. Overall, it seems like Word2Vec has the best performance by a slight edge, the second best being BERT, and the last Glove if we look into only the measure scores.

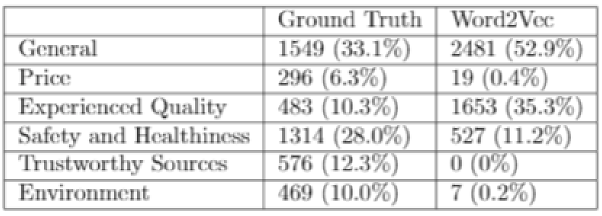
4.6.2 Class Distribution Comparison

The class distributions are shown in Figure 5 and Figure 6. By comparing different class distribution, BERT has the most unbalanced result as 92% data are in *General* Class. On the other hand, Word2Vec and GloVe also result in unbalanced but are more coherent with the distribution from the ground truth. However they both take high percentages in *Experienced Quality* class. As in ground truth class distribution, *Experienced Quality* class only takes 10.3% of the results, but the two word embeddings Word2Vec and GloVe take 35.3% and 44.6%. Another observation is that Word2Vec and GloVe almost do not predict *Trustworthy Sources* class at all, where BERT predicts 0.4% data as this class.

To summarize the observations, we could say that Word2Vec and Glove seem to have better performance than BERT by comparing different class distributions. However, the accuracy performance need to be further discussed.

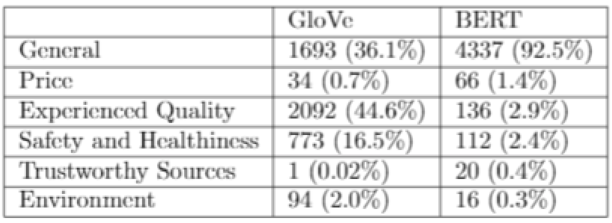
4.6.3 Qualitative Analysis Comparison

As we want to evaluate the actual performance of each word embedding, we evaluate some samples from predicted *Price* class as shown in Figure 7. The first four rows represent positive samples and the last three rows represent negative samples.



	Ground Truth	Word2Vec
General	1549 (33.1%)	2481 (52.9%)
Price	296 (6.3%)	19 (0.4%)
Experienced Quality	483 (10.3%)	1653 (35.3%)
Safety and Healthiness	1314 (28.0%)	527 (11.2%)
Trustworthy Sources	576 (12.3%)	0 (0%)
Environment	469 (10.0%)	7 (0.2%)

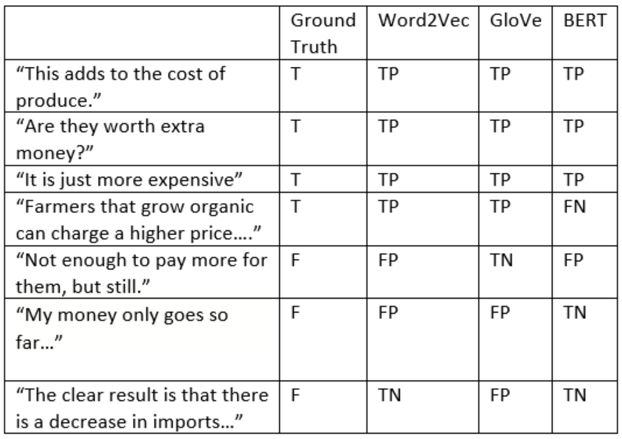
Figure 5 Class distribution of ground truth and Word2Vec prediction with the best micro f1 score



	GloVe	BERT
General	1693 (36.1%)	4337 (92.5%)
Price	34 (0.7%)	66 (1.4%)
Experienced Quality	2092 (44.6%)	136 (2.9%)
Safety and Healthiness	773 (16.5%)	112 (2.4%)
Trustworthy Sources	1 (0.02%)	20 (0.4%)
Environment	94 (2.0%)	16 (0.3%)

Figure 6 Class distribution of GloVe and BERT prediction with the best micro f1 score

As each of the word embedding models has some wrong predictions and the table is just a small part of the whole *Price* class, we can not really tell which word embedding perform the best. However we can see that they do not always have the same prediction on the same input.



	Ground Truth	Word2Vec	GloVe	BERT
"This adds to the cost of produce."	T	TP	TP	TP
"Are they worth extra money?"	T	TP	TP	TP
"It is just more expensive"	T	TP	TP	TP
"Farmers that grow organic can charge a higher price…."	T	TP	TP	FN
"Not enough to pay more for them, but still."	F	FP	TN	FP
"My money only goes so far…"	F	FP	FP	TN
"The clear result is that there is a decrease in imports…"	F	TN	FP	TN

Figure 7 Sample results that predicted as *Price* class

5 Conclusion and Future Work

Our investigations into the effectiveness of the teacher-student co-training as described in the paper[5] demonstrated that aspect detection can be done through weak supervision by providing only a small set of seedwords. As for the effect of using different seed word extraction methods, we find that Clarity Scoring Function gives the most relevant seed words for the respective aspects in Organic dataset. As intuitive as it is, we also found that providing more seed words lead to a more accurate/powerful classification model.

The research question as to the efficacy of the word embeddings remains an open question as each of the three models performed in a very comparable manner with Word2Vec performing slightly better among the three. The lowest micro f1 score was provided by GloVe model whereas BERT model had the worst prediction class distribution among the three. More research along different metrics is to be done to effectively distinguish between the models and their performance.

The unavailability of the annotated German dataset hampered the development around the German test-case. In future work, we plan to extend our framework to test on the German dataset. Additionally, memory constraints afffected the fine-tuning of GloVe vectors. Availability of higher resources/better methods in the future may also be considered in future additions and improvements to this project.

References

[1] Stefanos Angelidis and Mirella Lapata. Summarizing opinions: Aspect extraction meets sentiment prediction and they are both weakly supervised. *arXiv preprint arXiv:1808.08858*, 2018.

[2] Avrim Blum and Tom Mitchell. Combining labeled and unlabeled data with co-training. In *Proceedings of the Eleventh Annual Conference on Computational Learning Theory*, page 92–100. Association for Computing Machinery, 1998.

[3] Jacob Devlin, Ming-Wei Chang, Kenton Lee, and Kristina Toutanova. Bert: Pre-training of deep bidirectional transformers for language understanding. *arXiv preprint arXiv:1810.04805*, 2018.

[4] Nicholas Dingwall and Christopher Potts. Mittens: an extension of GloVe for learning domain-specialized representations. In *Proceedings of the 2018 Conference of the North American Chapter of the Association for Computational Linguistics: Human Language Technologies, Volume 2 (Short Papers)*, pages 212–217. Association for Computational Linguistics, 2018.

[5] Giannis Karamanolakis, Daniel Hsu, and Luis Gravano. Leveraging just a few keywords for fine-grained aspect detection through weakly supervised co-training. *arXiv preprint arXiv:1909.00415*, 2019.

[6] Da Kuang, Jaegul Choo, and Haesun Park. Nonnegative matrix factorization for interactive topic modeling and document clustering. In *Partitional Clustering Algorithms*, pages 215–243. Springer, 2015.

[7] Bing Liu and Lei Zhang. A survey of opinion mining and sentiment analysis. In *Mining text data*, pages 415–463. Springer, 2012.

[8] Tomas Mikolov, Kai Chen, Greg Corrado, and Jeffrey Dean. Efficient estimation of word representations in vector space. *arXiv preprint arXiv:1301.3781*, 2013.

[9] Jeffrey Pennington, Richard Socher, and Christopher D Manning. Glove: Global vectors for word representation. In *Proceedings of the 2014 conference on empirical methods in natural language processing (EMNLP)*, pages 1532–1543, 2014.

[10] Alex Ratner, Stephen Bach, Paroma Varma, and Chris Ré. Weak supervision: the new programming paradigm for machine learning. *Hazy Research. Available via https://dawn. cs. stanford. edu//2017/07/16/weak-supervision/. Accessed*, pages 05–09, 2019.

[11] Hassan Saif, Miriam Fernández, Yulan He, and Harith Alani. On stopwords, filtering and data sparsity for sentiment analysis of twitter. 2014.

[12] Xinjie Zhou, Xiaojun Wan, and Jianguo Xiao. Representation learning for aspect category detection in online reviews. In *Proceedings of the AAAI conference on artificial intelligence*, volume 29, 2015.