Learning multilingual topics through aspect extraction from monolingual texts

Johannes Huber
Otto-von-Guericke-Universität Magdeburg
& TrustYou GmbH, Munich
johannes@trustyou.net

Myra Spiliopoulou
Otto-von-Guericke-Universität Magdeburg
Faculty of Computer Science
Working Group "Knowledge Management & Discovery"
myra@ovgu.de

Abstract

Texts rating products and services of all kind are omnipresent on the internet. They come in various languages and often in such a large amount that it is very time-consuming to get an overview of all reviews. The goal of this work is to facilitate the summarization of opinions written in multiple languages, exemplified on a corpus of English and Finnish reviews. To this purpose, we propose a framework that extracts aspect terms from reviews and groups them to multilingual topic clusters.

For aspect extraction we work on texts of each language separately. We evaluate three methods, all based on neural networks. One of them is supervised, one unsupervised, based on an attention mechanism and one a rule-based hybrid method. We then group the extracted aspect terms into multilingual clusters, whereby we evaluate three different clustering methods and juxtapose a method that creates clusters from multilingual word embeddings with a method that first creates monolingual clusters for each language separately and then merges them.

We report on our results from a variety of experiments, observing the best results when clustering aspect terms extracted by the supervised method, using the k-means algorithm on multilingual embeddings.

Tiivistelmä

Tekstejä, jotka arvostelevat erilaisia tuotteita ja palveluja löytyy kaikkialta netistä. Niitä on usealla kielellä ja niin monia, että on hyvin aikaa vievää luoda yleiskuva kaikista arvosteluista. Tämän työn päämäärä on helpottaa objektiivisen yhteenvedon luomista mielipiteistä, jotka ovat kirjoitettu useammalla kielellä, mikä työssä on havainnollistettu niin englannin- kuin suomenkielisellä aineistolla. Tähän tarkoitukseen työ ehdottaa viitekehystä joka poimii aspektisanat arvosteluista ja ryhmittää ne monikielisiin aiheklustereihin.

Proceedings of the fifth Workshop on Computational Linguistics for Uralic Languages, pages 154–183,
Tartu, Estonia, January 7 - January 8, 2019.

Poimimivaihe tehtiin erikseen molempien kielien kohdalla. Vertailemme kolmea metodia, jotka kaikki käyttävät neuroverkkoja. Ensimmäinen metodi on valvottu ja toinen on hybridi, sääntöihin perustuvan poimimisen sekä valvotun opettamisen välimuoto. Viimeinen metodi on valvomaton ja perustuu huomiointimekanismiin. Sen jälkeen poimitut aspektitermit ryhmitetään monikielisiin aiheklustereihin. Testaamme kolme eri klusterointialgoritmia ja vertailemme kahta eri metodia monikielisten klustereiden tekemiseen: yksikielisten sanaedustumisen kohdistamista yhteen vektoritilaan sekä erikseen yksikielisille klustereille ryhmittämistä ja jälkeenpäin klustereiden kohdistamista.

Raporttina voimme muun muassa kertoa saaneemme parhaimmat tulokset poimimalla aspektisanat valvotulla metodilla ja ryhmittämällä k-means algoritmin monikielisten sanaedustumisen kanssa.

1 Introduction

Texts expressing opinions about products are becoming important for a constantly increasing number of people. From 2011 to 2017, the percentage of customers in the United States that reads online reviews to determine if a business is good or bad at least occasionally has grown from 71% to 93% (Anderson, 2017). Summarizing these reviews objectively can help customers in their choice of a product. As only about 40% of internet content is in English (Pimienta et al., 2009), analyzing reviews also in other languages appears vital to give a full picture of opinions about an entity. In this work, we propose a framework that derives aspect terms from reviews written in different languages and then summarizes them into multilingual topics.

Aspect term extraction is a part of aspect-level sentiment analysis (ALSA). ALSA is able to provide a detailed analysis of opinions conveyed in a text by extracting the sentiment expressed towards each mentioned aspect. For example, given the sentence ”the waitress was friendly”, it should extract a positive sentiment towards the aspect ”waitress”. As creating summaries or statistics on these aspects alone would result in a lot of clutter, it is beneficial to group semantically similar words into ”topics”; for example, aspect terms ”waitress”, ”waiter” and ”bartender” could form a topic ”staff”.

A survey by Schouten and Frasincar (2016) provides an overview about ALSA, but reports nearly exclusively on research on English corpora. Indeed, the vast majority of research on Sentiment Analysis and also natural language processing (NLP) in general has been done with English. Crosslingual NLP tries to utilize resources from a source language (generally English) for application on another target language. This is of advantage for languages where resources (in our work: opinions) are very rare. Multilingual NLP rather combines resources from different languages to analyze content written in them (Utt and Padó, 2014). In our work, we adhere to the second approach, in order to make full use of documents available in each of the languages under consideration.

We address the question ”How can we extract mono-lingual aspect terms from reviews in different languages and then combine them into multilingual topics that describe the multilingual corpus?”. We evaluate different methods for both the aspect extraction and the clustering step, focusing on ways of reducing human involvement and automating the learning process with minimal human input.

As proof of concept of our approach, we study a corpus containing English and Finnish reviews of restaurants. These two languages belong to unrelated families (Indoeuropean vs Uralic), differ in the amount of available resources (Finnish resources

are sparse), and are linguistically very different: English is a language with comparatively little morphology, while Finnish is an agglutinative language with very rich morphology (Pirkola, 2001). Our results show that multilingual topics can be extracted for even so different languages, making full use of the resources available in each language.

This study is organized as follows. In the next section we discuss relevant research advances. In section 3 we describe our framework, its components and the mechanisms used to evaluate each component. Our experiments and results are presented in section 4. In section 5 we discuss our findings. The last section concludes the paper with and outlook on future work and extensions.

2 Current Research State

2.1 Aspect extraction

Schouten and Frasincar in their 2016 survey classify the approaches to aspect detection into five different general methods: frequency-based, syntax-based, based on supervised learning, based on unsupervised learning and hybrids between the aforementioned.

2.1.1 Unsupervised approaches

The unsupervised methods presented in the survey are mostly based on Latent Dirichlet Allocation (LDA), in variants to make it work on the aspect level, which is far finer grained than the document level LDA was designed for. For example, the relatively recent Amplayo and Song (2017) combine LDA with Biterm Topic Models. Asnani and Pawar (2017) use more or less default LDA but combines it with semantic information from a multilingual dictionary, which allows them to extract aspects from code-mixed text, in this case social media content written in a combination of Hindi and English.

There are also some unsupervised approaches not based on LDA. Schouten et al. (2018) presents an unsupervised method based on association rule mining on co-occurrence frequency data. Also very recently, Dragoni et al. (2018) use NLP methods to get grammar dependencies and POS of a sentence and use rules based on that information to extract aspects from real-time stream data. A different approach is taken in the paper by He et al. (2017), which is based on an attention model, and which is the unsupervised method we decided to evaluate in this work.

2.1.2 Supervised approaches

For supervised approaches, we only examined methods that do not require the definition of a static set of aspects, but see the problem as a sequence labeling task.

State-of-the-art approaches train Deep Neural Networks on word embeddings for aspect term extraction. Usually general purpose word embeddings are used, however Pham et al. (2017) focuses on training word embeddings specifically for aspect extraction. The first deep learning based method was Poria et al. (2016), which uses word embeddings enriched by POS tags to surpass all previous approaches to aspect extraction significantly. The Xu et al. (2018) builds up on that, using double embeddings, which in this case means a combination of general-purpose embeddings with domain specific ones. Other recent supervised approaches using deep learning include Luo et al. (2018), which uses embeddings acquired from a bidirectional dependency tree

network to train a classifier, and Li et al. (2018), which uses an attention-based model in combination with selective transformation to not only extract aspect terms but also the corresponding opinion words. The last three papers mentioned report very similar performance values. We used (Xu et al., 2018) as the supervised method for our experiments.

2.1.3 Hybrid approach

A hybrid system, combining unsupervised extraction with training a Neural Network is described in Wu et al. (2018). We also tested this approach in our experiments.

2.2 Multi- and Crosslingual NLP

Multi- and crosslingual NLP has been dominated by methods utilizing word embeddings in the last years. A relatively recent *not* embedding-based approach is described by Täckström et al. (2012), where crosslingual word clusters are used to transfer models to predict linguistic structure between languages. These semantic clusters are built first for one language in the way described in by Brown et al. (1992) and then combined by projection.

A survey on crosslingual word embeddings was compiled by Ruder et al. (2017). It suggests a taxonomy of training crosslingual word-embedding models, which is classifying them based on the training data required: parallel or just comparable, aligned on word, sentence or document level.

Dufter et al. (2018) claim the current state-of-the-art model for sentence-aligned methods, called "concept induction". A parallel corpus is taken as input and used to induce dictionary graphs. From the dictionary graphs, concepts and words-concept pairs are then induced from the dictionary graph. Finally, embeddings are learned from the word-concept pairs using the standard Word2Vec method (Mikolov et al., 2013).

Word-aligned models usually use both bi- or multilingual dictionaries and big monolingual corpora in the target languages. The method we used for our experiments was presented by Joulin et al. in 2018. It is based on creating a restrained mapping between the two target vector spaces using the entries from the bilingual dictionary as anchor points. Artetxe et al. (2017) use a similar approach, but focus on reducing the amount of training data required by a self-learning method.

Some recent papers present ways to align word embeddings without any training data at all. Lample et al. (2018) and Zhang et al. (2017) use adversial training for this. In adversial training, two networks are used to provide training signals for each other. Lample et al. (2018) is also remarkable for presenting *MUSE*, an evaluation framework for multilingual embeddings that we also used as the basis for some of our experiments. Hoshen and Wolf (2018) instead of adversial training use iterative matching methods and Alvarez-Melis and Jaakkola (2018) see the task as an optimal transport problem and use the Gromov-Wasserstein distance to align embedding spaces.

3 Framework for extracting aspect terms and learning multilingual topics

Figure 1 shows the components of the system and the data passed between them, together with task descriptions for the more complex components. The tasks and

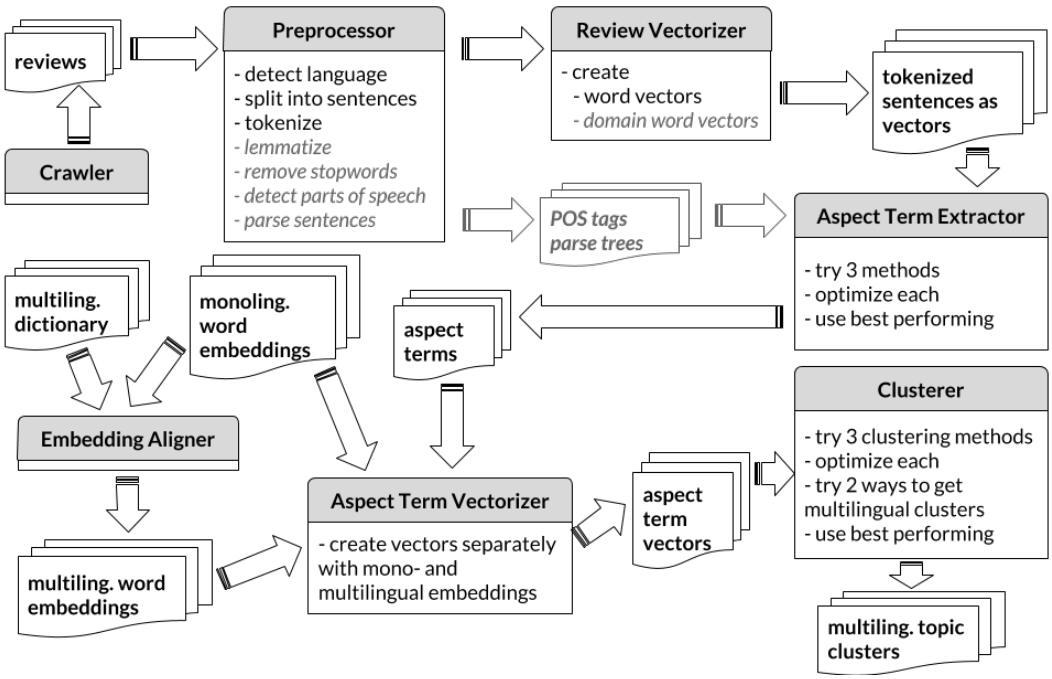

Figure 1: Architecture of system components and passed data

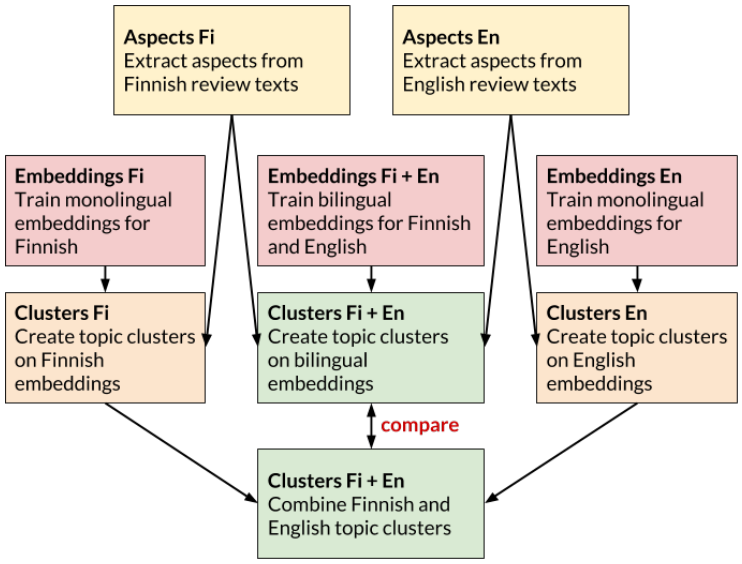

Figure 2: Multilingual workflow of the system

data printed in greyed out, italic letters are only relevant for some of the methods evaluated for the Aspect Term Extractor. Figure 2 shows the workflow of the system, with the focus put on exhibiting which parts are monolingual and which multilingual.

First, reviews are crawled from internet sources, then preprocessed and vectorized as required by the method to be tested in the Aspect Term Extractor. Each method is optimized, the best performing one is used to extract the set of aspect terms required for the next steps. This happens independently for each language.

In parallel, a set of multilingual word embeddings is created by aligning monolingual word embeddings of the target languages, using a dictionary that maps words between the languages to train the alignment.

The Aspect Term Vectorizer uses the aspect terms as well as both the monolingual and the previously obtained multilingual embeddings to create aspect term vectors - for each aspect terms once with the monolingual and once with the multilingual embeddings.

These vectors are then clustered to topics. From the monolingual embeddings, monolingual clusters are formed and then combined; from the multilingual ones, the multilingual clusters are formed directly. The performance of both the two ways of getting multilingual clusters and of the three different clustering methods that are evaluated is compared. The best performing method is used to create the final set of multilingual topic clusters.

In the following subsections, the different components are outlined in detail.

3.1 Preparing review texts for classification

This section describes the "Preprocessor" and "Review Vectorizer" components.

The language of each review is identified using *langid.py* (Lui and Baldwin, 2012), reviews not belonging to one of the target languages are filtered. Reviews are split into sentences with the PUNKT sentence segmenter (Kiss and Strunk, 2006), using the default NLTK (Bird et al., 2009) model for the respective language. The Penn Treebank tokenizer (Marcus et al., 1993) was then used to split the sentence into tokens.

To improve the performance of both sentence segmentation and tokenization, if a fullstop, colon, exclamation mark or quotation mark was directly followed by an upper-case character, a space was inserted in between. Without this step, the sentence segmenter would usually not split the sentence in case of this relatively common error.

The reason to choose these relatively simple methods over more sophisticated ones is their applicability to many languages: PUNKT was specifically designed as a multilingual method and the tokenizer is using relatively simple regular expressions that work for most languages.

The tokens in each sentence are then vectorized by assigning them a word embedding from a general-purpose dataset of pretrained word embeddings.

Some of the methods tested in the Aspect Term Extractor require additional preprocessing steps or use additional data in their vectors. These method-dependent preprocessing steps are outlined in the method descriptions.

3.2 Aspect term extraction

This section describes the "Aspect Term Extractor" component. The task of this component is to extract aspect terms from review sentences. We see aspects in the sense of the SemEval Task 2016/5 (Pontiki et al., 2016), called there "opinion target expression":

> [...] an explicit reference (mention) to the reviewed entity of the [entity-aspect] pair. This reference can be a named entity, a common noun or a multi-word term [...]

In other words, in a phrase expressing an opinion towards an aspect of the reviewed entity, it is the term explicitly referring to the aspect. It can be

- a named entity, like *"My **Sprite** was lukewarm when I got it."*
- a common noun, like *"The **bartender** excelled in his job."*
- a multi-word term, like *"I loved the **meat balls with mashed potatoes**!"*

To represent the positions of aspects in a sentence, sequence labelling with labelling space $\{B, I, O\}$ is performed. That means that each word in a sentence gets assigned a tag: either O when it is not part of an aspect, B when it is a single-word term or the first word of a multi-word term or I when it is a later word of a multi-word aspect term. An example:

The/O chicken/B wings/I were/O tasty/O and/O their/O price/B moderate./O

We evaluated the supervised method by Xu et al. (2018), the hybrid method by Wu et al. (2018) and the unsupervised method by He et al. (2017) which we outlined in the previous chapter. All methods evaluated operate on a sentence level, so each sentence is seen as independent.

3.2.1 Supervised method: Xu et al. (2018)

The supervised method was presented by Xu et al. in the paper "Double Embeddings and CNN-based Sequence Labeling for Aspect Extraction".

Their main contribution is to use what the authors call "double embeddings" as features. Double embeddings are concatenated general and domain-specific word embeddings. The general embeddings are trained on a huge, general dataset, the domain-specific embeddings on a dataset matching the target domain as exactly as possible. Labeled review sentences represented by these embeddings are used to train a relatively simple convolutional neural network.

This method requires the creation of domain-specific word vectors in the Review Vectorizer. We used this method without any changes. Besides in preprocessing, no changes were required to use this method for Finnish.

3.2.2 Hybrid method: Wu et al. (2018)

Wu et al. presented the hybrid method we are evaluating in the paper "A hybrid unsupervised method for aspect term and opinion target extraction".

The basic idea is to create training data for a deep-learning classifier by using some linguistic rules to create possible candidates, which are then filtered according to their domain correlation. The trained neural network is used to improve the domain correlation filter, which results in better training data for the next iteration, and so on. We evaluate the system performance either using the filtered candidates as the prediction or using the neural network to predict the tags.

While we tried to implement the model following the description in the paper as closely as possible, we had to make some adjustments in the selection of initial aspect candidates and the domain correlation filtering. The general architecture and the classifier remain unchanged.

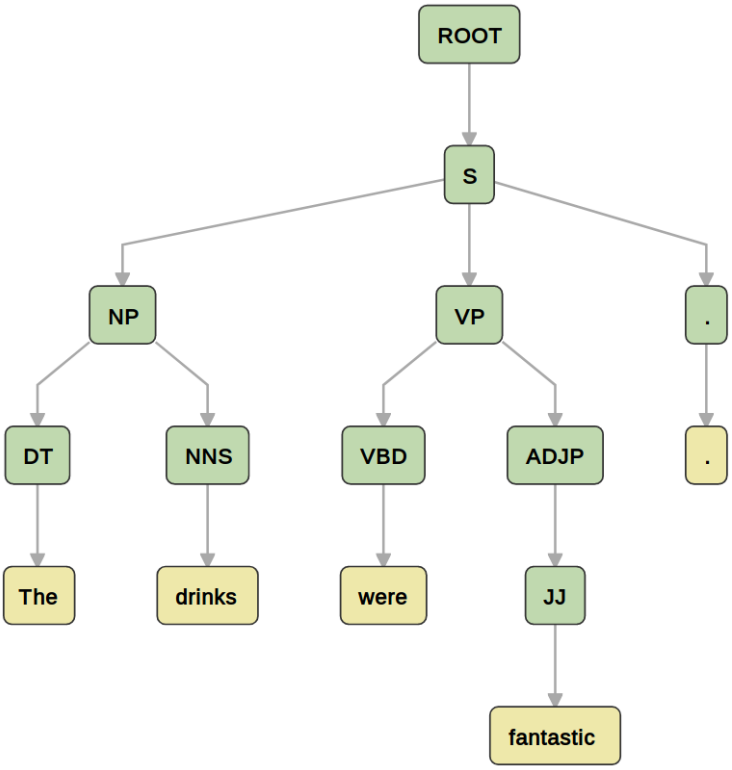

Figure 3: Constituency parse tree

English Since the original paper doesn't describe the initial creation of candidates to the last detail and we slightly diverged from it, we are presenting the full process we implemented in the following paragraphs.

First, for each sentence a parse tree like the one displayed in figure 3 is created using the NTLK (Bird et al., 2009) interface to the Stanford CoreNLP phrase constituency parser (Manning et al., 2014). All subtrees with "NP" (noun phrase) as the root node that have a height of 3 or 4 are extracted from the tree. A height of 3 means the level directly over part-of-speech (POS) tags and manages to capture mainly simple phrases like those consisting of just a noun; a NP with a height of 4 could for example be two nouns connected by a conjunction.

The noun phrases are filtered to only keep those which either include an adjective, adverb or a modal verb themselves or have a verbal phrase which includes one of these parts of speech as their right neighbor. Additionally, noun phrases that have a verb in base form as their left neighbour are kept, this is meant to capture phrases like "try the sushi".

Next, we remove overlapping phrases, which can exist because we initially picked phrases of both height 3 and 4. This is done as follows: The tree of height 4 is discarded if all of its NP-subtrees are also included in the set of trees eligible at this point. If that is not the case, the subtrees of height 3 are discarded.

From the remaining phrases, we remove all words that are not nouns of some kind or connectors. If connectors are at the beginning or the end of a phrase, they are, as the next step, also removed. The remaining words are seen as the final aspect term

candidates and passed to the domain correlation filter.

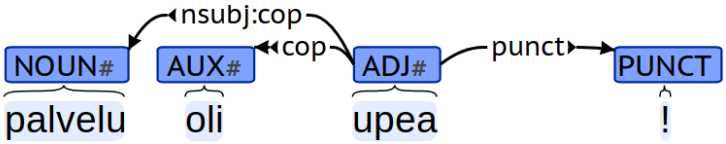

Figure 4: Dependency parse with TurkuNLP

Finnish For Finnish, we had to take a different approach to extracting initial candidates, since no phrase constituency parser is available for that language. However, a good dependency parser, created by TurkuNLP group (Kanerva et al., 2018) exists, which yields POS-tags and depencies between words as shown in figure 4. That allowed us to extract candidates in the following way:

We first extract all nouns that either are modified by an adjective directly or had a copula verb relation to any word. An example for a sentence with a copula verb relation is ”palvelu oli upea” (”the service was great”), where ”oli” (”was”) is the copula verb. In a second step, if another noun is either part of a compound with a noun chosen in the first step or a nominal modifier of such, this other noun is added to the term. The resulting noun phrases are the final aspect candidates.

Different than English, Finnish is a language with extremely many variations of each word. Because of that, we experimented with doing domain correlation filtering on lemmas instead of the word forms: all forms of a lemma should have the same domain correlation. The lemmas are created as part of the TurkuNLP parsing pipeline. As it is not guaranteed that a meaningful word embedding exists for a lemma, we represent a lemma by the embedding of its most frequent form. We always used lemmas to create the set of ”domain words” against which every other word is compared in order to decide on its domain correlation. For the sentences used to train the classifier, we did experiments both with and without lemmatizing each word in them.

Besides that, the architecture is the same as for English and as described in the paper.

3.2.3 Unsupervised method: He et al. (2017)

The unsupervised method was presented in the paper ”An Unsupervised Neural Attention Model for Aspect Extraction” by He et al.. This method does not train any classifier, but instead tries to compute representations for a set of topics[1], in the same vector space as the word embeddings. These topics are not predefined, but the number of topics is a fixed hyperparameter. The topic embeddings can be interpreted by looking at the closest words around them, which should be a set of semantically related words. They are learned by first determining a sentence representation using an attention model to determine the weight of each word in it and then reducing the error of recreating this sentence from the topic embeddings. As the model extracts the words that are most important both for a topic and in a sentence, this can be used to extract aspect words as well.

[1]In the paper, the authors are using the term ”aspects” for what we call ”topics”. We adopted this to our terminology for consistence.

We used the method basically as suggested in the original paper. It requires lemmatization, stopword removal and part-of-speech tagging as additional preprocessing steps.

The focus of the paper is on forming coherent topic clusters from words and not on extracting aspect terms in our sense. The clusters presented in their paper therefore contain also many words that are not aspect terms in the sense desired for this work. However, since the goal of the attention model is to put focus on words that have a high importance both towards the sentence and towards the aspect, the vectors representing the weight of each word in a sentence create a good basis to extract aspect terms from them. We did so by simply using all nouns whose weight is over a specified threshold as aspect terms.

3.2.4 Baseline values

To put the performance of the three models in relation, the following simple baseline values are given:

- Taking all nouns as aspect terms; if multiple nouns follow in a row, the later nouns get an I tag.
- Using the aspect term candidates extracted for the hybrid method as described in section 3.2.2 directly.

3.2.5 Evaluation

As evaluation metrics for the aspect term extractor, precision, recall and F1-value are computed by comparing the output of a classifier with labeled data. A correctly identified aspect term is seen as a correct match, if it is not correctly identified it's a false one. This means that correctly set O tags do not increase the precision or recall. This method is described for example in Tjong Kim Sang and Buchholz (2000).

For example, if one of the methods would return the following tag sequence:

B O B I O O O B

and the ground truth is the following sequence:

B O B O O B B O

the precision would be 0.333, since only one of three detected matches is correct; the recall would be 0.250, since only one of four actual matches is found. The F1 score would be 0.286, as it is the harmonic mean between precision and recall.

As seen in the example, only full matches are seen as correct, partial matches are treated the same as wrong matches.

3.3 Training Multilingual Embeddings

This section describes the "Embedding Aligner" component. The goal of training multilingual word embeddings is to create embeddings for words from multiple languages in the same vector space. These embeddings should have the same properties across languages as embeddings for one language, i.e. similar words should appear closely together in the vector space.

To obtain multilingual embeddings, we use pretrained monolingual embeddings and use the method described by Joulin et al. (2018). It works as follows: First, a

linear mapping between the vectors of the words that are in the training dictionary is learned. The mapping is optimized by minimizing the average of the loss between the mapped vectors of the source language and the vectors of the target language. The used loss function is based on the cosine similarity between the two vectors and symmetrically the average cosine similarity between one vector and the k nearest neighbours of the other vector (with k being a hyperparameter). The mapping is restrained to be orthogonal, which leads to the distances between the vectors being preserved from the original monolingual embeddings.

We used the reference implementation provided by the authors completely unchanged, also using their recommended hyperparameters.

3.4 Term Clustering

In this section, we describe the components "Aspect Term Vectorizer" and "Clusterer".

The goal of the clustering task is to create groups of words that are semantically coherent, i.e. describe the same topic. We are evaluating three different clustering methods: k-means (in the k-means++ variant (Kanungo et al., 2004)), Affinity Propagation (Frey and Dueck, 2007) and the attention-based method by He et al. described in section 3.2.3. K-means and Affinity Propagation are widely used general clustering algorithms that have been successfully used for the clustering of word embeddings (e.g. Kutuzov (2018); Cha et al. (2017); Suárez-Paniagua et al. (2015)), while the attention-based method was specifically developed for our target task.

We first vectorize the aspect terms, using either the multilingual embeddings obtained from the aligner or the monolingual embeddings directly. Then we try the different clustering methods and ways of obtaining multilingual clusters and use the best performing one to create the desired multilingual clusters.

3.4.1 K-means

The k-means algorithm is based on determining k centroid points, each of which defines a cluster as the points that are closer to it than to any other centroid. The distance used is the euclidean distance. The centroids are initialized randomly and then in each iteration chosen as the mean of the points in the centroid's cluster. After updating the centroids, the assignments of points to clusters are recomputed. This procedure is repeated until updating the centroids no longer leads to changes in the clustering, i.e. until the algorithm converged. The k-means++ variant we used differs from original k-means in the initialization of centroids, which is optimized for faster convergence.

3.4.2 Affinity propagation

Affinity propagation is an algorithm that does not require specifying the number of clusters. It uses the concept of passing messages between points in order to determine which points are chosen as *exemplar points*. Each non-exemplar point is assigned to exactly one exemplar point and all points that belong to the same exemplar form a cluster.

The algorithm starts with considering all points as possible exemplars. In each iteration, messages are passed between points in two steps, *responsibility* and *availability*. Responsibility values are sent towards candidate exemplar points, indicating how likely a point considers the candidate to be its exemplar. Availability values are the reverse, being sent from the candidate exemplars towards other datapoints and

reflecting how suitable the exemplar would be as an exemplar for a point. Responsibility is calculated using the distance between the two points and takes both availability of the previous iteration and distance towards other possible exemplars for the point into account. This results in the responsibility value being lowered if there are many other good exemplar candidates. Availability values are calculated from the responsibility of an exemplar candidate with itself and with other points. This results in a higher value if many other points see the candidate as suitable to be an exemplar.

The algorithm terminates either after a set number of iterations or when the number of clusters hasn't changed for some iterations. After termination, for each point the exemplar candidate with the highest value for summed availability and responsibility is chosen as its exemplar. If this candidate is the point itself, that point is an exemplar.

There are two main hyper-parameters for this algorithm to tune: The damping factor is used to determine the extent to which responsibility and availability values are updated over iterations, i.e. how big the impact of the value in the previous iteration is. A higher damping value indicates a higher weight to the previous value. The second hyper-parameter is the preference, which indicates how likely each point is chosen to be a exemplar. A higher preference value correlates with more clusters being created.

3.4.3 Attention based clustering

The method for attention-based clustering has already been described in section 3.2.3. Each topic embedding defines a cluster, with each point being assigned to the topic embedding closest to it.

3.4.4 Assigning multi-word terms to a cluster

All of the clustering methods are done on word embeddings, assigning each embedding of an aspect term to a cluster. This works well for single-word aspect terms, since their embeddings are either directly in the embedding set or can be inferred from subword information (Bojanowski et al., 2017). For aspects terms consisting of more than one word, this is not possible. While in theory it is possible to train embeddings for n-grams (Zhou et al., 2017), this would require training word embeddings specificially for our dataset and wouldn't allow us to use pretrained embeddings.

Therefore, we use the following approach: We train the clusterings on only single-word terms. Then, we check for each word in the multi-word term which cluster it would be assigned to. The full multi-word term is assigned to the cluster most of the words in it are assigned to. In case there isn't one cluster assigned more often than all others, we assign one of the most frequent clusters randomly for affinity propagation. For k-means and the attention-based method, we use the distances between the words in the term and the centroid (resp. topic embedding) of the cluster as a tie-breaker; the cluster with the lowest distance gets assigned.

3.4.5 Merging monolingual to multilingual clusters

To merge mono-lingual clusters of the different languages to multilingual ones, we used the bilingual dictionary also used for creating the multilingual clusters. For each cluster in the source language we checked in which clusters the translations of the words in it are in the clustering of the target language. The cluster gets merged with the cluster containing most translations.

3.4.6 Evaluation

Clusterings are evaluated against a pre-defined clustering. While this is in some ways slightly against the original purpose of dynamically creating topic clusters without pre-defining the set of topics, it appears to be the only way of providing an objective evaluation. In order to maintain the sense of dynamic clustering, we are mainly interested in seeing if clusters contain only terms belonging to one topic and not so much if there are clusters that could maybe be merged. To give an example, we would like to penalize if *bartender* and *salmon steak* are in the same cluster, since they very clearly do not belong to the same topic. We do not care much though if *salmon steak* and *beef tenderloin* are in the same cluster or not, since this is just a matter of how fine-grained the topic clustering is.

In order to meet this evaluation goal, we only define very few, broad clusters to evaluate against and see the *homogeneity* score (Rosenberg and Hirschberg, 2007) as our primary evaluation metric. Homogeneity is maximized when all clusters contain only elements from one ground-truth class, with 1 being the maximum and 0 the minimum value. Homogeneity strongly prefers fine-grained clustering over coarse grained ones; in the most extreme case, if a clustering would contain one cluster for each datapoint, homogeneity would be maximised. We therefore don't accept too fine-grained clusterings and also report the complementing score, *completeness*, which is maximized when all ground-truth classes contain only elements of one cluster.

This evaluation method is based on the way He et al. (2017) are evaluating their results. The main difference is that they manually assign clusters to ground-truth classes, which we avoid.

4 Experiments and Results

4.1 Implementation

All code is written in Python 3. We use PyTorch (Paszke et al., 2017) as the framework for all deep learning methods except for the unsupervised aspect extraction method, which uses TensorFlow (Abadi et al., 2016). The clustering methods use the implementations from the Scikit-learn framework (Pedregosa et al., 2011).

We implemented the crawling, preprocessing and vectorization components ourselves. For the other components, we used existing implementations of the tested methods as the base when they were available and extended and adjusted them to fit into our architecture. The only method completely implemented from scratch is the hybrid aspect extraction method, as no reference implementation has been published for it.

4.2 Aspect Term Extraction

In this section we present the experiments done in the Aspect Term Extractor, which are aimed at finding the best method of extracting aspect terms from review sentences.

4.2.1 Dataset

This subsection describes the datasets used for the experiments. Which data was used for which experiment is explained in detail in the subsections for each method.

English For English, we used SemEval 2016 Task 5 (Pontiki et al., 2016) as the annotated dataset. This dataset consists of 2674 sentences, of which 2000 are considered training and 674 testing data. Some of these sentences are marked as "out of scope" in the dataset and not annotated, so these were removed here. 2579 sentences remain. For the hybrid and unsupervised methods, an additional corpus of 75000 restaurant reviews, which consist of 368551 sentences, was used. These reviews are a random selection of reviews provided by *TrustYou GmbH*[2], which is a company focusing on review management for hotels and restaurants . The reviews were collected from different public sources, including *TripAdvisor*, *Google*, *OpenTable*, *Facebook* and *Zomato*.

We use the pretrained word embeddings provided by the GloVe project (Pennington et al., 2014), as this embedding set was used also in the original experiments for the supervised method (Xu et al., 2018). It was trained on the CommonCrawl corpus, a general-purpose text corpus that includes text from several billion web pages; the GloVe embeddings were trained on 840 billion tokens. The GloVe set includes embeddings for 2.2 million words, the embeddings have 300 dimensions. As domain-specific embeddings for the supervised method, we use the embedding set provided by the authors, which is 100-dimensional and was trained with *FastText* (Bojanowski et al., 2017) on a dataset provided by *Yelp*.

Finnish For Finnish, it was more difficult to obtain a sizable corpus of restaurant reviews. We ended up crawling the page *eat.fi*, a website for reviews of restaurants in Finland. After filtering out all reviews written in a language different than Finnish with *langid.py* (Lui and Baldwin, 2012), the obtained dataset consists of 71730 reviews, or 346144 sentences. 250 of these reviews, consisting of 1076 sentences, were labelled manually by the author. A subset of 70 reviews was additionally labelled by a native speaker; no major discrepancies in annotation were discovered. As general word embeddings, we use the Finnish word embeddings provided by *FastText* (Grave et al., 2018), which are also 300 dimensional and were trained on both CommonCrawl and Wikipedia data, together about 6 billion tokens. The provided dataset contains embeddings for exactly 2 million words, but also includes sub-word information that allows inferring embeddings for unknown words (Bojanowski et al., 2017). We trained domain-specific embeddings ourselves with *FastText* on the full dataset of restaurant reviews. We used the default parameters of *FastText* to train 100-dimensional vectors.

4.2.2 Baseline

Table 1 shows the baseline values for the aspect extraction task. For both languages, these values were computed on the complete annotated datasets, consisting of 2579 sentences for English and 1076 sentences for Finnish.

	English		**Finnish**	
	Nouns	Rules	Nouns	Rules
Precision	0.204	0.375	0.355	0.520
Recall	0.802	0.563	0.822	0.554
F1	0.430	0.450	0.496	0.537

Table 1: Baseline values for English and Finnish

[2]www.trustyou.net

4.2.3 Supervised

Datasets For the supervised method, the annotated data for Finnish was split to use 80% of the data for training and 20% for testing. This amounts to 216 testing and 860 training sentences. 128 of the training sentences were held out for choosing the best model and optimizing hyperparameters. For English, we used the SemEval 2016 Task 4 (Pontiki et al., 2016) dataset as suggested. After filtering "out-of-scope" sentences, that's 642 sentences for testing and 1937 for training. 150 training sentences were used for optimization.

English We attempted to recreate the English results from the paper (which uses the same dataset), but ended up with slightly worse values: With exactly the same hyperparameters, the model got an F1-Value of 0.724 (average over 5 runs), compared to the 0.747 reported in the paper. It is however to note that the performance deviation between runs is relatively high, with values ranging from 0.713 to 0.731 in the 5 runs. The best of the 5 runs had a precision of 0.674 and recall of 0.802. These values are all created with the evaluation tool provided by SemEval, which calculates slightly different values than our evaluation tool. With our evaluation script, the F1 value of the best run is 0.730, the averaged one 0.722. Since the difference between the values is very small and a detailed analysis of the differences is made difficult by the SemEval tool not being Open Source, we omit a further investigation. All other values reported in this paper are created with our evaluation script.

Finnish For Finnish, we tested different learning rates and dropout values. The results are displayed in table 2. All values are the average of three independent runs. The other hyperparameters were kept the same as in the paper.

Dropout	40			55			70		
Learning Rate	10^{-5}	10^{-4}	10^{-3}	10^{-5}	10^{-4}	10^{-3}	10^{-5}	10^{-4}	10^{-3}
Precision	0.597	0.628	**0.698**	0.591	0.614	0.669	0.000	0.607	0.685
Recall	0.672	0.724	0.719	0.626	0.749	0.732	0.000	**0.781**	0.729
F1	0.632	0.672	**0.707**	0.608	0.675	0.699	0.000	0.683	0.706

Table 2: Results for different dropout and learning rate values in Finnish

The experiments show that the dropout rate has very little influence on the result. The learning rate however has a significant influence, with performance generally increasing with bigger learning rates, despite the high number of training iterations (200). In all experiments, recall was at least slightly higher than precision. The result for a dropout of 70 and a learning rate of 10^{-5} sticks out as the system in this case learned to always predict the label *O*. An explanation for this result could be the choice of the loss function: The negative log-likelihood is calculated for every possible target label, including *O*. With *O* being, naturally for this task, the by far most frequent label, a slight bias towards choosing it can be expected. This is however contrary to our evaluation method, for which always predicting *O* is the worst possible result. It is unclear why this happens only for this specific combination of parameters.

4.2.4 Hybrid

For the hybrid method, we trained the model with the full dataset (annotated and unannotated) for both English and Finnish and evaluated it on the annotated dataset. For English, we additionally did experiments where we used only the annotated dataset for both training and evaluation.

We used a mini-batch size of 64 for all experiments. This value is not given in the original paper, as well as the learning rate. The latter we optimized as explained in the following subsection.

English For English, we first did some experiments to determine good hyper- parameters using only the annotated data for training. Training for six iterations (updating the domain correlation filter and thereby the training data after each iteration) and ten epochs per iteration, we optimized separately the learning rate and the minimum correlation required to pass the correlation filter. All other hyper-parameters were kept as reported in the paper. Results for different learning rates can be found in table 3; we used 0.50 as the minimum correlation here. Table 4 shows results for different minimum correlation values, with the learning rate set to the best value found, 0.001. The columns in the section "Classifier" mean the performance of the trained classifier, the columns in the "Filter" section mean the performance when using the filtered aspect candidates as the prediction.

	Classifier			**Filter**		
Learning rate	0.0001	0.001	0.01	0.0001	0.001	0.01
Precision	0.263	0.299	0.266	0.390	0.390	**0.704**
Recall	0.668	**0.685**	0.582	0.333	0.331	0.086
F1	0.377	**0.416**	0.365	0.360	0.358	0.153

Table 3: Results for using different learning rates for English

	Classifier				**Filter**			
Min correlation	0.40	0.45	0.50	0.55	0.40	0.45	0.50	0.55
Precision	0.300	0.301	0.299	0.286	0.373	0.373	0.390	**0.508**
Recall	**0.725**	0.712	0.685	0.685	0.473	0.414	0.331	0.241
F1	**0.424**	0.423	0.416	0.404	0.417	0.393	0.358	0.327

Table 4: Results for using different correlation value cut-offs for English

The experiments show that the influence of the learning rate is again relatively big, similarly to the supervised method. On the other hand, changing the minimum correlation value to pass the filter has a quite low influence. Using a lower minimum correlation value slightly increases the recall and the F1 value, as the precision stays about constant.

Using the best values of these two experiments, we also used the full dataset, including both annotated and unannotated data, for training. We set the minimum frequency to be included into the set of domain words to 75. The result barely changed compared to training on only the annotated data: The classifier's precision slightly increased to 0.321, however recall fell to 0.626, resulting in an unchanged F1 value of

0.424. For using the output after the filtering step, the precision rose to 0.580, but with a significantly lower recall of 0.183, the F1 value dropped to 0.278.

Finnish Since the amount of annotated data available is significantly lower for Finnish than for English, we for Finnish only ran experiments using the full dataset, both annotated and unannotated, for training. Using the same hyperparameters as for training on the full English dataset, we got the results for Finnish shown in table 5.

	Classifier		**Filter**	
	Not lemmatized	Lemmatized	Not lemmatized	Lemmatized
Precision	0.391	0.367	**0.851**	0.678
Recall	**0.703**	0.657	0.196	0.471
F1	0.503	0.471	0.318	**0.556**

Table 5: Results for Finnish with and without lemmatization

4.2.5 Unsupervised

For the unsupervised, attention-based method we did most tests using the full dataset, containing both annotated and unannotated data, for training and evaluated the performance on the complete annotated dataset.

We kept all hyper-parameters as in the paper. We tested the influence of the numbers of created clusters on the performance, which turned out to be negligible, so we kept it at 14 (which is the number used in the paper).

For our modification of the method to obtain aspect terms, we had to introduce an additional hyper-parameter, which is the minimum weight of a word to be used as an aspect term. The performance for different values, both for English and Finnish, can be seen in table 6.

	English				**Finnish**			
Min weight	0.05	0.10	0.15	0.20	0.05	0.10	0.15	0.20
Precision	0.346	0.415	0.433	**0.453**	0.409	0.471	0.489	**0.507**
Recall	**0.696**	0.555	0.510	0.462	**0.744**	0.588	0.546	0.507
F1	0.462	**0.473**	0.468	0.458	**0.528**	0.523	0.516	0.507

Table 6: Results for experiments with different minimum aspect weights

This shows a precision/ recall trade-off: The lower the minimum weight, the higher the recall but the lower the precision. This result proves that the attention of a word generally is correlated to the likelihood of it being a aspect term. However, since recall decreases stronger than precision increases, a lower minimum weight leads generally to a higher F1 score.

For English, we additionally tested the performance when training and testing on only the annotated dataset. The F1 value was for all weight-cutoffs two to three percentage points lower than when using the full dataset.

4.2.6 Comparison

Table 7 shows a performance comparison between the three different methods and the baseline.

	English				Finnish			
Method	Baseline	Superv.	Hybrid	Unsup.	Baseline	Superv.	Hybrid	Unsup.
Precision	0.375	0.669	0.300	0.415	0.520	0.698	0.678	0.409
Recall	0.563	0.784	0.725	0.555	0.554	0.719	0.471	0.744
F1	0.450	0.722	0.424	0.473	0.537	0.707	0.556	0.528

Table 7: Summary of the best results for all methods

We see that the supervised method works best with a significant margin. The hybrid and unsupervised methods are at about the level of the rule-based baseline. Results for Finnish and English are comparable, with slightly better results for English with the supervised method and for Finnish with the other methods.

4.3 Multilingual embeddings and clusterings

In this section, we present the experiments evaluating the different ways of clustering and of creating multilingual clusters. This concerns primarily the "Clusterer" component, with additionally the "Embedding Aligner" playing a role in the experiments with multilingual embeddings.

4.3.1 Datasets

We use mainly the same datasets as for the aspect extraction task. The English labeled data from SemEval already contains category information, assigning each aspect term one of the classes *ambiance*, *drinks*, *food*, *location*, *restaurant* and *service*. The *restaurant* category is used for terms describing the restaurant in general and such that don't match one of the other categories. For the Finnish labeled data, we manually assigned each unique aspect term to one of these six classes. The English dataset contains 874 unique aspect terms, the Finnish dataset 623.

Evaluation was done for all experiments with the full labeled datasets. We did experiments both with training the clusters on only the labeled datasets and with training them on the 5000 most frequent single-word aspect terms extracted by the best performing aspect extraction method from the full datasets.

For both English and Finnish, we use as word embeddings the pretrained *FastText* embeddings, which were trained on CommonCrawl and Wikipedia data and include subword information. For Finnish, this is the same embedding set used as for the aspect extraction task, for English, it is different. The reason for this is that we wanted to have embeddings trained in the same way for both languages, since we assumed that this would improve performance for the creation of multilingual embeddings from them.

Both for creating multilingual word embeddings and for clustering we only worked with the embeddings of words actually required. This includes

- all unique aspect terms,

- the full vocabulary of our datasets, preprocessed as for the attention-based aspect extraction and clustering method (which is lemmatized and reduced to only include words that appear at least two times in the corpus),
- words from the evaluation datasets.

In total, this results in 25808 words for English and 28327 words for Finnish. We did this mainly because the script to create multilingual embeddings is very memory-intensive and was not possible to run with the full embedding sets on our machines. Also, this procedure allowed us to utilize the sub-word information of the FastText embeddings and create embeddings for all words in our vocabulary, also such that are not part of the pretrained set.

4.3.2 Multilingual embeddings

We used the default parameters for training multilingual embeddings and ran the training for 25 iterations. We tested if there is a performance difference between aligning English embeddings to the Finnish embedding space or the other way round. The performance was slightly better when treating English as the target embedding space and aligning the Finnish embeddings into it, so we went with this direction.

4.3.3 Clustering

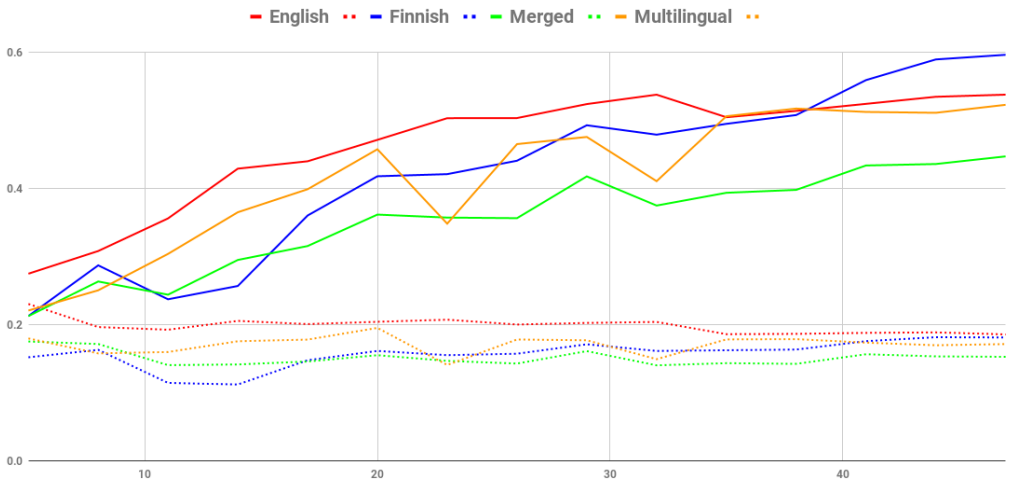

Figure 5: Performance of k-means for different k values. Straight lines: Homogeneity, dotted lines: Completeness

k-means We tested clusterings from 8 to 50 clusters in steps of 3. Figure 5 shows homogeneity and completeness scores for English and Finnish monolingual clusterings, as well as for multilingual clusterings, either based on multilingual embeddings or on merged clusters. As expected, for all of these measurements homogeneity values increased with an increasing number of clusters, up to around 0.60 for Finnish and 0.55 for English. Completeness values stayed more or less constant at a low value of around 0.2.

Using multi-lingual embeddings results in homogeneity scores up to 0.53, the best score when merging monolingual clusters is about 0.46.

	Finnish	English	Multilingual	Merged
Annotated only	0.493	0.524	0.475	0.418
Full dataset	0.366	0.493	0.381	0.380

Table 8: Homogeneity values for clusters trained on either the full or only the the annotated dataset

Table 8 shows homogeneity values for cluster size 29, which seemed like a good trade-off between a not too large number of clusters and a good homogeneity score. It shows in comparison the results for training clusters on only aspect terms from the annotated dataset and on also using the terms extracted by the supervised algorithm from the full dataset. The difference is between 2 and 13 percentage points, with the smallest difference for the monolingual English clusters and the biggest difference for monolingual Finnish clusters. This performance trend is also valid for other cluster sizes. For English, clusters trained on the full dataset work sometimes even slightly, up to 3 percentage points, better than those trained on only the annotated dataset. For Finnish, the performance is always at least 5 percentage points lower.

Affinity Propagation Initial experiments showed that the damping factor had nearly no influence on the resulting performance, so we set it to 0.9, the value suggested by the authors of the original paper (Rosenberg and Hirschberg, 2007).

The preference value however does have a very significant influence on the number of clusters created and therefore on our performance measurements.

We tested preference values from -42 to -6 in steps of 3, after that in steps of 1 until 0. We discarded any clustering with more than 30 clusters. Table 9 shows the best homogeneity performance for each experiment setup, together with the number of clusters created and the chosen preference value.

	Annotated data only				**Full dataset**			
	En	Fi	ML	Mgd	En	Fi	ML	Mgd
Homogeneity	0.364	0.461	0.344	0.295	0.433	0.393	0.323	0.347
Completeness	0.150	0.158	0.142	0.123	0.171	0.138	0.119	0.145
Clusters	22	30	25	22	30	29	32	30
Preference	-3	-3	-6	-3	-21	-24	-42	-21

Table 9: Best results for clustering with affinity propagation

The experiments show that the number of clusters created with the same preference value is strongly dependent on the amount of data. Comparing the optimal preference values for the experiments with multilingual embeddings, which contain about twice the amount of data, to the other experiments, we see that the required preference value to get a similar number of clusters is also about twice as small.

Another thing to notice is that using the full dataset for training increases the performance of the clustering for English and the merged clusters; it is to notice however that the number of clusters with the full data is slightly larger for these experiments.

Also it can be seen that merging monolingual clusters works worse than using multilingual embeddings when training on only the annotated dataset, but slightly better when using the full dataset. Completeness scores are about constant for all experiments at values around 0.15.

Attention Table 10 shows results for clustering with the attention model from He et al. for different numbers of clusters. We ran tests for 14, 28 and 42 different clusters, 14 being the number chosen in the original paper. All other hyper-parameters we set to the best results from the aspect extraction experiments, see section 4.2.5 for details. We created the topic clusters from the complete review dataset.

	14	28	42
English	0.218	0.258	0.125
Finnish	0.171	0.169	0.165
Multilingual	0.107	0.053	0.023
Merged	0.134	0.174	0.076

Table 10: Homogeneity values for clustering with the attention-based model, using different numbers of clusters

Different to the other methods, homogeneity scores don't generally increase with more clusters here; the results for 42 clusters are significantly worse than for 14 or 28 clusters across all setups. 28 clusters work best for English and merged clusters, for Finnish and when using multilingual embeddings 14 clusters work better. Using multilingual embeddings results in significantly worse values than merging clusters, English performs better than Finnish.

Comparison For all experiments we see that clustering with k-means works, for a similar amount of clusters, better than the other methods. The difference to using affinity propagation is relatively small, the attention based method works a lot worse. This means that the simplest and fastest method works best in our experiments.

For creating multilingual clusters, we see better results when using multilingual embeddings compared to creating monolingual clusters and merging them. Clustering with multilingual embeddings achieves nearly the same performance values as monolingual clusterings. We see that the two languages have generally similar performance in the monolingual experiments, with, depending on the setup, one or the other language performing slightly better. For English and when merging clusters, clustering on the full, automatically extracted set of aspect terms results in about the same performance as clustering on only the manually annotated terms. For Finnish and when using multilingual embeddings, using the full dataset yields worse results.

5 Discussion

5.1 Aspect Term Extraction

In the section about aspect term extraction, we showed that the supervised deep learning method is beating the performance of methods that don't require annotated data by a relatively big margin. This can partly be explained by the nature of the other methods we used. The hybrid method is mainly based on filtering candidates from

the rule-based system. However, as we achieve recall values of only about 0.55 with the rule-based extraction, it doesn't appear like focusing on removing candidates from this set is the right approach to increase results. On the other hand, for using the filtered rule-based output to train a classifier, the precision isn't good enough, as can be seen with the best performance value for the English hybrid model, which achieves relatively high recall of 0.73 but a precision of only 0.3.

For the unsupervised model, reasons are similar: Since the model gives a weight to every word in the sentence, not just those that would possibly be aspect terms in our sense, we had to add some simple rules to get aspect terms comparable to the other methods. While these rules on their own achieve a recall of about 80% (and a very low precision), the method can under no circumstances find aspect terms that don't match these rules. While the attention model does appear to be meaningful in some way, the precision gains from filtering candidates is lower than the loss of recall, which results in a relatively poor overall performance.

The supervised model however achieves a good F1 score of about 0.7 both for English and Finnish. This is especially remarkable with the strict criterion of only treating exact matches as correct. There have been shared tasks (e.g. Wojatzki et al. (2017)) that also counted aspect terms as correct when they were only overlapping with a ground truth term. A subjective look at the terms extracted by the model appears to confirm that the percentage of matches that a human would consider "okay" is significantly above 70%. Worth noting is also that the performance is about equal for English and Finnish, despite the very different language structure of English and Finnish and the significantly lower amount of training data for Finnish.

5.2 Clustering and multilingual embeddings

We showed that clustering with k-means yields better results than the other methods. The reason for the attention based method to work badly is most likely that the topic centers it creates are not generally meaningful for the task we evaluate. This is due to this method creating clusters on the full reviews, not only on the extracted aspect terms. While in theory the attention mechanism should put weight only on the words representing aspects, this doesn't appear to always work well in practice. Looking at the topic clusters the method created on the full dataset in the best performing experiment (English, 28 clusters), we find for example one cluster containing mainly first names and one cluster containing predominantly positive adverbs. While there also are some clusters that look very good, like one containing mostly pasta dishes, these cannot save the overall bad performance of the method when clustering aspect terms.

It is not clear why affinity propagation gives worse results than k-means. Previous works generally report better results for the former, also when working with textual data, for example Guan et al. (2011). However, Kutuzov (2018), who also clustered word embeddings, reports that the performance for his experiments was sometimes better with k-means and sometimes with affinity propagation, so it seems to be highly dependent on the data used.

The good performance when creating clusters on the set of terms extracted with the supervised method from the first task further proves its quality.

We showed that by using multilingual embeddings, we can achieve results similar to monolingual clusterings. Investigating the created clusters in detail, we can find that while many clusters are nicely merged including related terms from both languages, there are also some clusters that only include words from one language. This

shows that there is still further room for improvement in the task of creating multilingual embeddings, either by tuning hyper-parameters or by testing or developing new methods.

5.3 Extensibility to other languages

We showed that supervised aspect extraction methods work significantly better than unsupervised ones, which means training data has to be created for each new language to be added. However, we also show that about 1000 annotated sentences are enough to train a well-performing model. It may well be possible to further reduce this number by using active learning or transfer learning.

The method used for training multilingual embeddings was designed to work for more than two languages; its authors show that aligning 28 languages into the same vector space still works well on their evaluation tasks (Joulin et al., 2018). However, since we noticed several clusters that only contained embeddings from one language, the performance on our clustering task is likely to reduce by some extent. Besides that, clustering would work the same as for two languages though. If clustering with k-means, the number of clusters could just be kept constant, for affinity propagation, the *preference* value would have to be adjusted for the additional data.

To summarize, our method is extensible, but some manual work to integrate another language would be required.

6 Conclusion

6.1 Summary

In our work, we presented a framework to extract aspect terms from monolingual reviews and cluster them to multilingual topics. We showed that for aspect term extraction, the supervised method we tested worked significantly better than the hybrid and unsupervised methods, which did not manage to exceed the performance of our baselines. We showed that for the supervised method, performance for English and Finnish is about equal, without any language-specific adjustments made.

For the clustering subtask, the best performing method was the simplest one we tested, k-means. We showed that when using multilingual embeddings, the performance of the clustering is just slightly worse compared to clustering only Finnish or only English terms monolingually. We also showed that the results when clustering on only annotated aspect terms are only slightly better than when clustering on the set of aspect terms obtained from the supervised aspect extraction method.

6.2 Future work

6.2.1 Technical extensions

There are some smaller, technical improvements that could be done to potentially improve the results in our experiments, mainly in the clustering part. While the homogeneity values for clusters created with multilingual embeddings already are close to the performance of monolingual clusters, we still got some clusters that only included words from one language, indicating further potential for improvements of the alignment process. Potential areas for improvements could be extended training

dictionaries (for example by using *PanLex* (Kamholz et al., 2014)) or a more extensive search for the best hyper-parameters.

Another point for improvement of the clustering method is the evaluation method we chose. While using the classification categories from the SemEval data provided an objective truth to measure against and focusing on the homogeneity score allowed for finer-grained clusterings, we still can't really say which number of clusters yields the actually best clustering for the task. Determining which is the best clustering is a very subjective task, which points to a more detailed manual annotation of the resulting clusters being required.

6.2.2 Conceptual extensions

Other directions of improvement are of larger scale. In the last months and years, progress in the field of deep learning has been very rapid. Many of the new developments could also be applied to the tasks of this paper, especially to the part about aspect term extraction. The supervised model which yielded the best performance is based on a relatively simple and straightforward neural network. It is likely that network architectures better suited for the task exist, but manually trying them is extremely time intensive. Methods like *ENAS* (Pham et al., 2018) optimize the architecture search, for example by sharing trained parameters between related architectures. The authors claim that using their method is 1000 times less computationally expensive than trying different architectures without optimization. Also other aspects in the suggested model could potentially be further improved, for example variational, learned dropout rates have shown better results than static ones (Molchanov et al., 2017).

For clustering, we found that one of the most simple clustering methods, k-means, performs better than the more sophisticated methods we tried. Clustering with k-means has several weaknesses, like not being able to handle noise points and expecting non-overlapping clusters of similar sizes. However, especially when clustering the aspect terms extracted automatically, noise points have to be expected in our dataset; also the assumption of similar-sized clusters can not be made, since especially the "food" category is far bigger than the others. This indicates that there should be clustering methods that would be better suited for our problem than k-means. There are many additional methods to cluster high-dimensional data that could be tried. An overview is provided in Kriegel et al. (2009).

For real-world applications to make use of our work, mainly two changes appear to be necessary: For one, the analysis and summarization of reviews is usually desired on the level of the entity they refer to, in our case the restaurant. This is necessary to provide a basis of comparison between the reviews for the different restaurants. Also, it would probably be required to choose an representative name for each topic cluster, so that for example a sentiment score could be displayed for each topic instead of having to list all the aspect terms in this topic. With these extensions, a system to dynamically detect and summarize the most relevant topics for a restaurant could be built. Our work has hopefully provided the basis for that.

Acknowledgments

This work is inspired by the project OSCAR "Opinion Stream Classification with Ensembles and Active Learners" (funded by the German Research Foundation); Myra

Spiliopoulou is principal investigator for OSCAR.

References

Martin Abadi, Paul Barham, Jianmin Chen, Zhifeng Chen, Andy Davis, Jeffrey Dean, Matthieu Devin, Sanjay Ghemawat, Geoffrey Irving, Michael Isard, Manjunath Kudlur, Josh Levenberg, Rajat Monga, Sherry Moore, Derek G. Murray, Benoit Steiner, Paul Tucker, Vijay Vasudevan, Pete Warden, Martin Wicke, Yuan Yu, and Xiaoqiang Zheng. 2016. Tensorflow: A system for large-scale machine learning. In *12th USENIX Symposium on Operating Systems Design and Implementation (OSDI 16)*. pages 265–283. https://www.usenix.org/system/files/conference/osdi16/osdi16-abadi.pdf.

David Alvarez-Melis and Tommi Jaakkola. 2018. Gromov-wasserstein alignment of word embedding spaces. In *Proceedings of the 2018 Conference on Empirical Methods in Natural Language Processing*. Association for Computational Linguistics, Brussels, Belgium.

Reinald Kim Amplayo and Min Song. 2017. An adaptable fine-grained sentiment analysis for summarization of multiple short online reviews. *Data & Knowledge Engineering* 110:54–67. https://doi.org/10.1016/j.datak.2017.03.009.

Myles Anderson. 2017. Local consumer review survey 2017. *BrightLocal* https://www.brightlocal.com/learn/local-consumer-review-survey/.

Mikel Artetxe, Gorka Labaka, and Eneko Agirre. 2017. Learning bilingual word embeddings with (almost) no bilingual data. In *Proceedings of the 55th Annual Meeting of the Association for Computational Linguistics (Volume 1: Long Papers)*. Association for Computational Linguistics, Vancouver, Canada, pages 451–462. https://doi.org/10.18653/v1/P17-1042.

Kavita Asnani and Jyoti D. Pawar. 2017. Automatic aspect extraction using lexical semantic knowledge in code-mixed context. *Procedia Computer Science* 112:693–702. Knowledge-Based and Intelligent Information & Engineering Systems: Proceedings of the 21st International Conference, KES-20176-8 September 2017, Marseille, France. https://doi.org/10.1016/j.procs.2017.08.146.

Steven Bird, Ewan Klein, and Edward Loper. 2009. *Natural Language Processing with Python*. O'Reilly Media, Inc., 1 edition.

Piotr Bojanowski, Edouard Grave, Armand Joulin, and Tomas Mikolov. 2017. Enriching word vectors with subword information. *Transactions of the Association for Computational Linguistics* 5:135–146. http://aclweb.org/anthology/Q17-1010.

Peter F. Brown, Peter V. deSouza, Robert L. Mercer, Vincent J. Della Pietra, and Jenifer C. Lai. 1992. Class-based n-gram models of natural language. *Comput. Linguist.* 18(4):467–479. http://dl.acm.org/citation.cfm?id=176313.176316.

Miriam Cha, Youngjune Gwon, and H. T. Kung. 2017. Language modeling by clustering with word embeddings for text readability assessment. In *Proceedings of the 2017 ACM on Conference on Information and Knowledge Management*. ACM, New York, NY, USA, CIKM '17, pages 2003–2006. https://doi.org/10.1145/3132847.3133104.

Mauro Dragoni, Marco Federici, and Andi Rexha. 2018. An unsupervised aspect extraction strategy for monitoring real-time reviews stream. *Information Processing & Management* https://doi.org/10.1016/j.ipm.2018.04.010.

Philipp Dufter, Mengjie Zhao, Martin Schmitt, Alexander Fraser, and Hinrich Schütze. 2018. Embedding learning through multilingual concept induction. In *Proceedings of the 56th Annual Meeting of the Association for Computational Linguistics (Volume 1: Long Papers)*. Association for Computational Linguistics, Melbourne, Australia, pages 1520–1530. http://aclweb.org/anthology/P18-1141.

Brendan J. Frey and Delbert Dueck. 2007. Clustering by passing messages between data points. *Science* 315(5814):972–976. https://doi.org/10.1126/science.1136800.

Edouard Grave, Piotr Bojanowski, Prakhar Gupta, Armand Joulin, and Tomas Mikolov. 2018. Learning Word Vectors for 157 Languages. In Nicoletta Calzolari (Conference chair), Khalid Choukri, Christopher Cieri, Thierry Declerck, Sara Goggi, Koiti Hasida, Hitoshi Isahara, Bente Maegaard, Joseph Mariani, Hélène Mazo, Asuncion Moreno, Jan Odijk, Stelios Piperidis, and Takenobu Tokunaga, editors, *Proceedings of the Eleventh International Conference on Language Resources and Evaluation (LREC 2018)*. European Language Resources Association (ELRA), Miyazaki, Japan.

Renchu Guan, Xiaohu Shi, Maurizio Marchese, Chen Yang, and Yanchun Liang. 2011. Text clustering with seeds affinity propagation. *IEEE Trans. on Knowl. and Data Eng.* 23(4):627–637. https://doi.org/10.1109/TKDE.2010.144.

Ruidan He, Wee Sun Lee, Hwee Tou Ng, and Daniel Dahlmeier. 2017. An unsupervised neural attention model for aspect extraction. In *Proceedings of the 55th Annual Meeting of the Association for Computational Linguistics (Volume 1: Long Papers)*. Association for Computational Linguistics, Vancouver, Canada, pages 388–397. https://doi.org/10.18653/v1/P17-1036.

Yedid Hoshen and Lior Wolf. 2018. Non-adversarial unsupervised word translation. In *Proceedings of the 2018 Conference on Empirical Methods in Natural Language Processing*. Association for Computational Linguistics, Brussels, Belgium.

Armand Joulin, Piotr Bojanowski, Tomas Mikolov, Hervé Jégou, and Edouard Grave. 2018. Loss in translation: Learning bilingual word mapping with a retrieval criterion. In *Proceedings of the 2018 Conference on Empirical Methods in Natural Language Processing*. Association for Computational Linguistics, Brussels, Belgium.

David Kamholz, Jonathan Pool, and Susan Colowick. 2014. Panlex: Building a resource for panlingual lexical translation. In *Proceedings of the Ninth International Conference on Language Resources and Evaluation (LREC-2014)*. European Language Resources Association (ELRA), Reykjavik, Iceland. http://www.aclweb.org/anthology/L14-1023.

Jenna Kanerva, Filip Ginter, Niko Miekka, Akseli Leino, and Tapio Salakoski. 2018. Turku neural parser pipeline: An end-to-end system for the CoNLL 2018 shared task. In *Proceedings of the CoNLL 2018 Shared Task: Multilingual Parsing from Raw Text to Universal Dependencies*. Association for Computational Linguistics, Brussels, Belgium, pages 133–142. http://www.aclweb.org/anthology/K18-2013.

Tapas Kanungo, David M. Mount, Nathan S. Netanyahu, Christine D. Piatko, Ruth Silverman, and Angela Y. Wu. 2004. A local search approximation algorithm for k-means clustering. *Computational Geometry* 28(2):89–112. Special Issue on the 18th Annual Symposium on Computational Geometry - SoCG2002. https://doi.org/10.1016/j.comgeo.2004.03.003.

Tibor Kiss and Jan Strunk. 2006. Unsupervised multilingual sentence boundary detection. *Computational Linguistics* 32(4):485–525. https://doi.org/10.1162/coli.2006.32.4.485.

Hans-Peter Kriegel, Peer Kröger, and Arthur Zimek. 2009. Clustering high-dimensional data: A survey on subspace clustering, pattern-based clustering, and correlation clustering. *ACM Trans. Knowl. Discov. Data* 3(1):1–58. https://doi.org/10.1145/1497577.1497578.

Andrey Kutuzov. 2018. Russian word sense induction by clustering averaged word embeddings. *CoRR* abs/1805.02258. http://arxiv.org/abs/1805.02258.

Guillaume Lample, Alexis Conneau, Marc'Aurelio Ranzato, Ludovic Denoyer, and Hervé Jégou. 2018. Word translation without parallel data. In *International Conference on Learning Representations*. https://openreview.net/forum?id=H196sainb.

Xin Li, Lidong Bing, Piji Li, Wai Lam, and Zhimou Yang. 2018. Aspect term extraction with history attention and selective transformation. In *Proceedings of the Twenty-Seventh International Joint Conference on Artificial Intelligence, IJCAI-18*. International Joint Conferences on Artificial Intelligence Organization, pages 4194–4200. https://doi.org/10.24963/ijcai.2018/583.

Marco Lui and Timothy Baldwin. 2012. langid.py: An off-the-shelf language identification tool. In *Proceedings of the ACL 2012 System Demonstrations*. Association for Computational Linguistics, Jeju Island, Korea, pages 25–30. http://www.aclweb.org/anthology/P12-3005.

Huaishao Luo, Tianrui Li, Bing Liu, Bin Wang, and Herwig Unger. 2018. Improving aspect term extraction with bidirectional dependency tree representation. *CoRR* abs/1805.07889. http://arxiv.org/abs/1805.07889.

Christopher D. Manning, Mihai Surdeanu, John Bauer, Jenny Finkel, Steven J. Bethard, and David McClosky. 2014. The stanford corenlp natural language processing toolkit. In *Association for Computational Linguistics (ACL) System Demonstrations*. pages 55–60. http://www.aclweb.org/anthology/P/P14/P14-5010.

Mitchell P. Marcus, Mary Ann Marcinkiewicz, and Beatrice Santorini. 1993. Building a large annotated corpus of english: The penn treebank. *Comput. Linguist.* 19(2):313–330. http://dl.acm.org/citation.cfm?id=972470.972475.

Tomas Mikolov, Kai Chen, Greg Corrado, and Jeffrey Dean. 2013. Efficient estimation of word representations in vector space. *CoRR* abs/1301.3781. http://arxiv.org/abs/1301.3781.

Dmitry Molchanov, Arsenii Ashukha, and Dmitry Vetrov. 2017. Variational dropout sparsifies deep neural networks. In Doina Precup and Yee Whye Teh, editors, *Proceedings of the 34th International Conference on Machine Learning*. PMLR, International Convention Centre, Sydney, Australia,

volume 70 of *Proceedings of Machine Learning Research*, pages 2498–2507. http://proceedings.mlr.press/v70/molchanov17a.html.

Adam Paszke, Sam Gross, Soumith Chintala, Gregory Chanan, Edward Yang, Zachary DeVito, Zeming Lin, Alban Desmaison, Luca Antiga, and Adam Lerer. 2017. Automatic differentiation in pytorch. In *Proceedings of the NIPS 2017 Autodiff Workshop*.

Fabian Pedregosa, Gaël Varoquaux, Alexandre Gramfort, Vincent Michel, Bertrand Thirion, Olivier Grisel, Mathieu Blondel, Peter Prettenhofer, Ron Weiss, Vincent Dubourg, Jake Vanderplas, Alexandre Passos, David Cournapeau, Matthieu Brucher, Matthieu Perrot, and Édouard Duchesnay. 2011. Scikit-learn: Machine learning in python. *The Journal of Machine Learning Research* 12:2825–2830. http://dl.acm.org/citation.cfm?id=1953048.2078195.

Jeffrey Pennington, Richard Socher, and Christopher D. Manning. 2014. Glove: Global vectors for word representation. In *Empirical Methods in Natural Language Processing (EMNLP)*. pages 1532–1543. http://www.aclweb.org/anthology/D14-1162.

Duc-Hong Pham, Anh-Cuong Le, et al. 2017. Fine-tuning word embeddings for aspect-based sentiment analysis. In *International Conference on Text, Speech, and Dialogue*. Springer, pages 500–508.

Hieu Pham, Melody Guan, Barret Zoph, Quoc Le, and Jeff Dean. 2018. Efficient neural architecture search via parameters sharing. In Jennifer Dy and Andreas Krause, editors, *Proceedings of the 35th International Conference on Machine Learning*. PMLR, Stockholmsmässan, Stockholm Sweden, volume 80 of *Proceedings of Machine Learning Research*, pages 4095–4104. http://proceedings.mlr.press/v80/pham18a.html.

Daniel Pimienta, Daniel Prado, and Álvaro Blanco. 2009. *Twelve years of measuring linguistic diversity in the Internet: balance and perspectives*. United Nations Educational, Scientific and Cultural Organization Paris.

Ari Pirkola. 2001. Morphological typology of languages for ir. *Journal of Documentation* 57(3):330–348. https://doi.org/10.1108/EUM0000000007085.

Maria Pontiki, Dimitris Galanis, Haris Papageorgiou, Ion Androutsopoulos, Suresh Manandhar, Mohammed AL-Smadi, Mahmoud Al-Ayyoub, Yanyan Zhao, Bing Qin, Orphée De Clercq, Veronique Hoste, Marianna Apidianaki, Xavier Tannier, Natalia Loukachevitch, Evgeniy Kotelnikov, Núria Bel, Salud Maria Jiménez-Zafra, and Gülşen Eryiğit. 2016. Semeval-2016 task 5: Aspect based sentiment analysis. In *Proceedings of the 10th International Workshop on Semantic Evaluation (SemEval-2016)*. Association for Computational Linguistics, San Diego, California, pages 19–30. http://www.aclweb.org/anthology/S16-1002.

Soujanya Poria, Erik Cambria, and Alexander Gelbukh. 2016. Aspect extraction for opinion mining with a deep convolutional neural network. *Knowledge-Based Systems* 108:42–49. New Avenues in Knowledge Bases for Natural Language Processing. https://doi.org/10.1016/j.knosys.2016.06.009.

Andrew Rosenberg and Julia Hirschberg. 2007. V-measure: A conditional entropy-based external cluster evaluation measure. In *Proceedings of the 2007 Joint Conference on Empirical Methods in Natural Language Processing and Computational Natural Language Learning (EMNLP-CoNLL)*. http://www.aclweb.org/anthology/D07-1043.

Sebastian Ruder, Ivan Vulić, and Anders Søgaard. 2017. A survey of cross-lingual embedding models. *CoRR* abs/1706.04902. http://arxiv.org/abs/1706.04902.

Kim Schouten and Flavius Frasincar. 2016. Survey on aspect-level sentiment analysis. *IEEE Trans. on Knowl. and Data Eng.* 28(3):813–830. https://doi.org/10.1109/TKDE.2015.2485209.

Kim Schouten, Onne van der Weijde, Flavius Frasincar, and Rommert Dekker. 2018. Supervised and unsupervised aspect category detection for sentiment analysis with co-occurrence data. *IEEE Transactions on Cybernetics* 48(4):1263–1275. https://doi.org/10.1109/TCYB.2017.2688801.

Vıctor Suárez-Paniagua, Isabel Segura-Bedmar, and Paloma Martı´nez. 2015. Word embedding clustering for disease named entity recognition. In *Proceedings of the Fifth BioCreative Challenge Evaluation Workshop*. pages 299–304.

Oscar Täckström, Ryan McDonald, and Jakob Uszkoreit. 2012. Cross-lingual word clusters for direct transfer of linguistic structure. In *Proceedings of the 2012 Conference of the North American Chapter of the Association for Computational Linguistics: Human Language Technologies*. Association for Computational Linguistics, Stroudsburg, PA, USA, NAACL HLT '12, pages 477–487. http://dl.acm.org/citation.cfm?id=2382029.2382096.

Erik F. Tjong Kim Sang and Sabine Buchholz. 2000. Introduction to the conll-2000 shared task: Chunking. In *Proceedings of the 2Nd Workshop on Learning Language in Logic and the 4th Conference on Computational Natural Language Learning - Volume 7*. Association for Computational Linguistics, Stroudsburg, PA, USA, ConLL '00, pages 127–132. https://doi.org/10.3115/1117601.1117631.

Jason Utt and Sebastian Padó. 2014. Crosslingual and multilingual construction of syntax-based vector space models. *Transactions of the Association for Computational Linguistics* 2:245–258. https://transacl.org/ojs/index.php/tacl/article/view/240.

Michael Wojatzki, Eugen Ruppert, Sarah Holschneider, Torsten Zesch, and Chris Biemann. 2017. Germeval 2017: Shared task on aspect-based sentiment in social media customer feedback. In *Proceedings of the GermEval 2017 – Shared Task on Aspect-based Sentiment in Social Media Customer Feedback*. Berlin, Germany, pages 1–12.

Chuhan Wu, Fangzhao Wu, Sixing Wu, Zhigang Yuan, and Yongfeng Huang. 2018. A hybrid unsupervised method for aspect term and opinion target extraction. *Knowledge-Based Systems* 148:66–73. https://doi.org/10.1016/j.knosys.2018.01.019.

Hu Xu, Bing Liu, Lei Shu, and Philip S. Yu. 2018. Double embeddings and cnn-based sequence labeling for aspect extraction. In *Proceedings of the 56th Annual Meeting of the Association for Computational Linguistics (Volume 2: Short Papers)*. Association for Computational Linguistics, Melbourne, Australia, pages 592–598. http://aclweb.org/anthology/P18-2094.

Meng Zhang, Yang Liu, Huanbo Luan, and Maosong Sun. 2017. Adversarial training for unsupervised bilingual lexicon induction. In *Proceedings of the 55th Annual Meeting of the Association for Computational Linguistics (Volume 1: Long Papers)*. Association for Computational Linguistics, Vancouver, Canada, pages 1959–1970. https://doi.org/10.18653/v1/P17-1179.

Zhihao Zhou, Lifu Huang, and Heng Ji. 2017. Learning phrase embeddings from paraphrases with grus. In *Proceedings of the First Workshop on Curation and Applications of Parallel and Comparable Corpora*. Asian Federation of Natural Language Processing, Taipei, Taiwan, pages 16–23. http://aclweb.org/anthology/W17-5603.