Proceedings of the 12th Conference on Language Resources and Evaluation (LREC 2020), pages 4974–4981
Marseille, 11–16 May 2020

Aspect On: an Interactive Solution for Post-Editing the Aspect Extraction based on Online Learning

Mara Chinea-Rios, Marc Franco-Salvador, and Yassine Benajiba
Symanto Research, Nürnberg, Germany
{mara.chinea, marc.franco, yassine.benajiba}@symanto.net

Abstract

The task of aspect extraction is an important component of aspect-based sentiment analysis. However, it usually requires an expensive human post-processing to ensure quality. In this work we introduce Aspect On, an interactive solution based on online learning that allows users to post-edit the aspect extraction with little effort. The Aspect On interface shows the aspects extracted by a neural model and, given a dataset, annotates its words with the corresponding aspects. Thanks to the online learning, Aspect On updates the model automatically and continuously improves the quality of the aspects displayed to the user. Experimental results show that Aspect On dramatically reduces the number of user clicks and effort required to post-edit the aspects extracted by the model.

Keywords: aspect extraction, online learning, post-editing, annotation, user interface

1. Introduction

Aspect extraction is the task of automatically extracting opinion targets in text (Liu, 2012). For instance, from *the service was excellent*, it aims to extract *service*. This task is a key component of aspect-based sentiment analysis (Hu and Liu, 2004), where the objective is to extract the sentiment polarity associated with each aspect.

The task of aspect extraction consists of two principal steps: i) extract all the aspect terms from a dataset, e.g. *cleaner*, *laundry*, *coffee*, *pasta*; ii) group the aspect terms into categories that represent the same aspect, e.g. [*cleaner*, *laundry*] and [*potatoes*, *pasta*] for the *cleaning* and *food* aspects, respectively. In addition, there is a post-processing step that might be done to ensure the quality of the aspects. It usually comprises a manual revision and edition of the extracted aspects and of the aspect terms annotated in a dataset. This step is specially important in production environments or for creating high quality aspect-based gold standards (Ganu et al., 2009; McAuley et al., 2012).

One of the most accessible ways to post-edit the output of a model is through a User Interface (UI) (Wang and Soong, 2009). This method improves efficiency and productivity by providing users with tools specifically designed for their task. Furthermore, UIs provided with guidelines reduce the expertise required by the annotators, which makes it easier to find qualified assets. In case we want to include an online learning to optimize the task, UIs then, become an even more important component.

Throughout this paper, we use the definition of online learning as a method in which data becomes available in a sequential order and is used to update our best predictor for future data at each step (Anderson, 2008). It is specially used in scenarios where it is computationally infeasible to train over the entire dataset or when the model continuously needs to adapt to new patterns of data. The continuous process of improving a model by means of user validations and editions is framed in that last scenario. The tasks of image processing (Chechik et al., 2010), machine translation (Denkowski et al., 2014; Ortiz-Martínez et al., 2010) and sentiment analysis (Li et al., 2010) demonstrated the ability of the technique to improve their models.

In this work we introduce Aspect On, an interactive solution based on online learning that allows users to post-edit and improve the quality of an automatic aspect extraction. Aspect On relies on a neural model to conduct the automatic aspect extraction. The UI combines two functionalities: i) visualize and edit the aspect inventory; and ii) annotate and edit the aspect terms of a given dataset. Aiming to reduce the user efforts, Aspect On uses online learning to update the model weights. In this way, it adjusts better to the user edits and improves the quality of the aspect term annotations that are still pending of review in the dataset.

Experimental results with two domains, laptops and restaurants, show that our solution dramatically reduces the mouse-action ratio required to post-edit the aspect extraction, and consequently the user effort. The results also show that the edits of the user serve to improve our aspect extraction model in a automatic fashion. In addition, the study of the minimum amount of edits needed to improve the model provides interesting insights.

The contributions of this paper are:

1. Demo a solution that tackles the core problem in the opinion mining industry of getting the most accurate and custom aspects at the lowest manual labor possible;

2. show how this solution is possible because Aspect On is able to employ the user feedback at the sentence level to tune a neural model;

3. present a UI that is designed specifically for this matter; and

4. share with the community a full blown solution to accelerate research in this domain.

The rest of the paper is structured as follows. In Section 2. we review the state of the art on unsupervised aspect extraction. Section 3. presents our aspect-extraction post-editing solution. In Section 4. we present our evaluation and results. Finally, we conclude the paper in Section 5..

2. Related Work

The task of aspect extraction gained popularity in the last fifteen years as it proved to be fundamental for document indexing and opinion mining. In general, we can find in the literature two main types of approaches: supervised and unsupervised. In this paper we focus on the later one, which has been the preferred option because it does not require labeled data.
Topic modeling techniques such as latent Dirichlet allocation (Blei et al., 2003) provided with a more automatic and flexible unsupervised aspect extraction[1]. Topic models represent topics as probability distributions over words, and dataset documents as probability distributions over topics. Several works where aspects are derived from topics have been published in the last years (Titov and McDonald, 2008; Brody and Elhadad, 2010; Zhao et al., 2010; Mukherjee and Liu, 2012). The vocabulary selection and filtering performed using Dirichlet (or similar) based distributions provides with adequate aspect term candidates. However, topics are not clusters that group all similar elements together: topics tend to be small and may keep similarities among them. In addition, He et al. (2017) pointed out that most topic models do not preserve topic coherence: they exploit word-document occurrence instead of word co-occurrence. The authors of that work aimed to tackle the weaknesses of topics models and proposed a new neural architecture to extract coherent aspects. Their Attention-based Aspect Extraction (ABAE) model is inspired by autoencoders. It uses word embeddings combined with an attention mechanism to represent the input text. Then it reconstructs that text representation by means of a linear combination of aspect representations. The matrix that contains these aspects needs to be initialized with information that facilitates the model to learn in the right direction. They used K-means (MacQueen and others, 1967) centroids. We observed that the resulting aspects are mostly coherent and of notable quality. However, they might not cover specific aspects or need some edits to satisfy certain needs, specially in production environments or for creating golds standards. In this work we build on ABAE and combine it with a UI and online learning to ease the process of post-edition of aspect and aspect term annotations. In Section 3. we detail the modifications applied to ABAE and how we used it for conducting context-aware text annotations.
Other unsupervised neural approaches for aspect extraction include the use of restricted Boltzmann machine models together with PoS features for joint sentiment and aspect extraction (Wang et al., 2015). More recently, Karamanolakis et al. (2019) proposed a neural approach that starts from aspects defined with few seed words, and uses a teacher-student architecture and an iterative co-training to extract the ultimate aspects. It should be noted that the seed words cannot be modified to update the model and, in consequence, refine or add new aspects. In addition, a shared drawback with some of the aforementioned approaches is the absence of clear mechanisms to use the model for conducting context-aware text annotations.

[1]For clarification, the words 'topic' and 'aspect' can be used interchangeably in a general setting. It is more correct, however, to use 'topic' when we are aiming at finding the most important themes in a document and 'aspect' when we refer to characteristics of products, i.e. product aspects, when we are doing opinion mining.

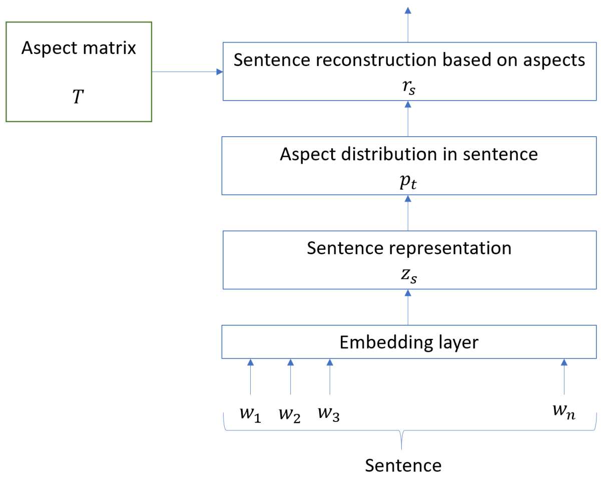


Figure 1: ABAE neural model architecture.

3. Aspect On

Aspect On consists of three main components: i) in Section 3.1. we describe the employed aspect extraction model; ii) the online learning method that updates the model to adjust better to the user edits and validations is described in Section 3.2.; and iii) in Section 3.3. we describe the UI of Aspect On.

3.1. Aspect Extraction Model

Our design and technical choices for the work we present in this paper are primarily guided by our goal to build a tool that caters to the everyday needs of consumer insights and marketing businesses. As we have previously mentioned, in such setting, it is key to have tools that are able to integrate the specifications of the project at hand without asking the user to annotate a large set of data. Using a tool that relies on seeds is not optimal either because it creates a tunnel vision effect for the user as it leaves out all the aspects that are not aligned with the seeds.
In this section, we describe the model that, to the best of our knowledge, is the most appropriate to satisfy these constraints. As well as how we extended it to be able to classify aspects at the word level and extract multi-word aspect terms.

3.1.1. Attention-Based Aspect Extraction Model

For the reasons mentioned above, we chose the Neural Attention-Based Aspect Extraction Model (ABAE) (He et al., 2017) as our core engine for aspect extraction. The only hyperparameter that needs to be predetermined by the user is the number of aspects K. Following, a neural model discovers the prominent K aspects in the provided text through a training process that learns how to reconstruct each sentence in the dataset.
More formally, we consider each dataset D as a set of sentences s_i where $i \in 1, 2, ...N$. Each sentence s_i is, in turn, a sequence of words w_j where $j \in 1, 2, ...n_{s_i}$. For each word

in the vocabulary forming D, we have a vector representation that can be obtained from a pre-trained embedding space.
As illustrated in Figure 1, the ABAE neural model has 4 layers:

1. An input layer that maps each word w_j in the sentence to its vector representation e_{w_j} .

2. A layer to compute the sentence vector representation. This representation is obtained through a weighted sum of the e_{w_j} values. Thus, we obtain z_s through the following formula:

$$\mathbf{z}_s = \sum_{j=1}^{n} a_j \mathbf{e}_{w_j} \quad (1)$$

   where a_j are the attention weights for each word in the sentence. To make sure a_j are positive and sum up to 1 they are computed as a softmax of previously obtained scores d_j:

$$a_j = \frac{\exp(d_j)}{\sum_{l=1}^{n_{s_i}} \exp(d_l)} \quad (2)$$

   These scores are the result of first a product among the transpose of e_{w_j}, a trainable weight matrix M and a vector y_s representing the average value for each dimension in our vector space in the sentence. Thus:

$$d_i = \mathbf{e}_{w_i}^{\top} \cdot \mathbf{M} \cdot \mathbf{y}_s \quad (3)$$

$$\mathbf{y}_s = \frac{1}{n} \sum_{i=1}^{n} \mathbf{e}_{w_i} \quad (4)$$

3. A layer that converts the sentence representation z_s into a probability distribution p_t over K aspects through a trainable set of weights W:

$$p_t = \text{softmax}(W \cdot z_s + b) \quad (5)$$

4. Finally, the last layer attempts to reconstruct z_s in a vector r_s by performing a sum weighted by p_t of all K elements of a matrix T:

$$r_s = T^{\top} \cdot p_t \quad (6)$$

where T is a matrix that lists the K points in the embedding space representing the most prominent aspects in the dataset. From this viewpoint, the ABAE network could be considered as implementing a search algorithm for the K points in the embedding space that can be used to form the representation of any sentence in the dataset. That is to say, once these K points are identified, we can represent any sentence as a weighted sum over them. This is different from most other aspect detection algorithms where the outcome is usually a lexicon of words representing each aspect. The main advantage of having K vectors instead is that we are not restricted to using string matching to annotate aspect terms in a sentence (as is the case where we have lexicons).
To train this model, a contrastive loss is used in a way that it encourages the model to reconstruct the sentence such as r_s is as similar to z_s as possible but, at the same time, as different as possible from a set of randomly selected sentences. Also, a regularization term is added to make sure that the K vectors in T are as distinct from each other as possible. For more details about the objective function, we would like to refer the reader to Section 3.3 of the original paper (He et al., 2017).

3.1.2. Our ABAE Extension

In our work, we extend the ABAE model in two ways:

1. The vector p_t computed at the third layer as shown by Equation 5 gives us an aspect distribution at the sentence level. However, since z_s is weighted sum of e_{w_j}, that means that the sentence and the word representations are in the same vector space (since there are no non-linear functions mapping the e_{w_j} to z_s). It follows then that we can use Equation 5 with the same set of parameters W to obtain an aspect distribution at the word level by simply replacing z_s with e_{w_j}. Thus, to get the aspect distribution, p_{t_j} for a word w_j represented with an embedding vector e_{w_j} we use the following formula:

$$p_{t_j} = \text{softmax}(W \cdot e_{w_j} + b) \quad (7)$$

   This is possible without retraining the model because the parameters W were trained to assign a aspect distribution to a vector that pertains to a vector space shared by both the word embeddings and the sentence representations. This allows us to perform context-aware aspect term annotations, i.e., to perform aspect term annotations considering the context of the term in the text. By this way we might have different aspects for same term depending on its context.

2. Using the attention scores a_j (see Equation 2) means that the model is using unigrams as unit of analysis. It is, however, the case that very frequently an aspect term is composed of more than one aspect word where each belongs to a different aspect category. For instance, the aspect term 'store events' is composed of two words where if we considered each separately we would need to account for both the aspects of 'SHOPS' and 'EVENTS', respectively, in our output. Considering that such output will be combined with other models to assign sentiment and/or other valence values to the aspects, the reader can easily see how this error will propagate to all the subsequent components of the pipeline. In our tool, we try to mitigate this issue by using a simple heuristic that concatenates multiple words as one aspect term if they: i) form a consecutive sequence in the sentence; and ii) have relevant attention values; and iii) belong to the same aspect. Finally, it is an aspect term if appears in the dataset more than the frequency aspect term threshold. The frequency aspect threshold is obtained empirically during the experimental process.

3.2. Online Learning Algorithm

As we described in the previous section, the ABAE model operates in a completely unsupervised fashion to discover the aspects that are manifest in the data. This offers a very cost efficient solution. However, since the model is uninformed about the business need in question, it will, as expected, provide results that ignore its nuances. The tool we present in this paper offers a solution to tune the results obtained from ABAE with a minimum of manual effort.
The intuition behind our online algorithm is that we want to request user feedback at the optimal point that offers the highest amount of information at the lowest amount of manual work to tune the model. We chose the second layer where we can compute the attention scores and the aspect distribution at the word level as such point. In other words, for a random sample of sentences provided by the user we can use the current model to annotate each word as i) whether it is an aspect word or not; and ii) the most likely aspect category in the case that it is (see Section 3.3. for how this is presented in the UI). When the user visualizes and corrects the incorrect annotations, the information is used to tune the neural model in two phases:

- **Phase 1:** We use the correction to tune the attention weights. To do so, we assign a probability q_j to each word, such as:

$$q_j = \begin{cases} \frac{1}{m}, & \text{if annotated as aspect term by the user;} \\ 0, & \text{otherwise.} \end{cases} \tag{8}$$

  where m is the number of words highlighted by the user as aspect terms. To tune the first layer of our model with these newly annotated sentences, we use a cross entropy loss function that can be expressed as:

$$l = \sum_k q_k . \log(a_k) \tag{9}$$

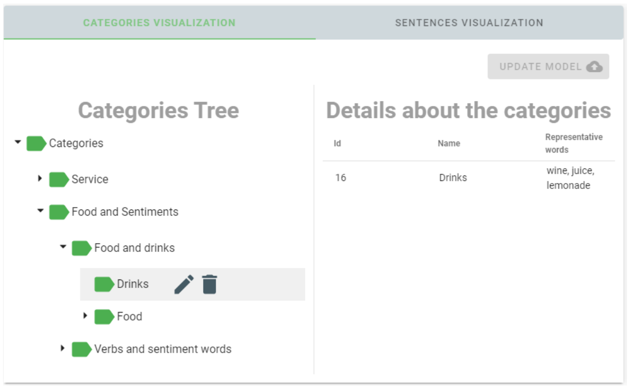
- **Phase 2:** we use the correction to tune layers 2-4 of the model. To achieve this goal we use the annotated sentences (by the user) as input of layer 2 and we tune the model with the same logic and loss function mentioned in Section 3.1.1.. The only difference, as we just mentioned, is that the input of layer 2 will be a sentence representation z_s^* built solely from the words annotated by the user as aspect terms instead of z_s which is computed as presented in Equation 1. This is possible by using the probabilities q_j presented in Equation 8 instead of the a_j scores computed by layer 1 to form the sentence representation z_s^*, thus the the formula to compute z_s^* is:

$$\mathbf{z}^*{}_s = \sum_{j=1}^{n} q_j \mathbf{e}_{w_j} \tag{10}$$

3.3. User Interface

Aspect On is implemented following a server-browser scheme. Therefore, its UI is accessible via a web browser.[2]

[2]The UI is implemented using CSS, JavaScript, and Vue.js (`https://vuejs.org/`).



Id	Name	Representative words
16	Drinks	wine, juice, lemonade

Figure 2: Aspect inventory. The left column corresponds to the hierarchical tree of the aspects. The right column displays the aspect name and its representative terms.

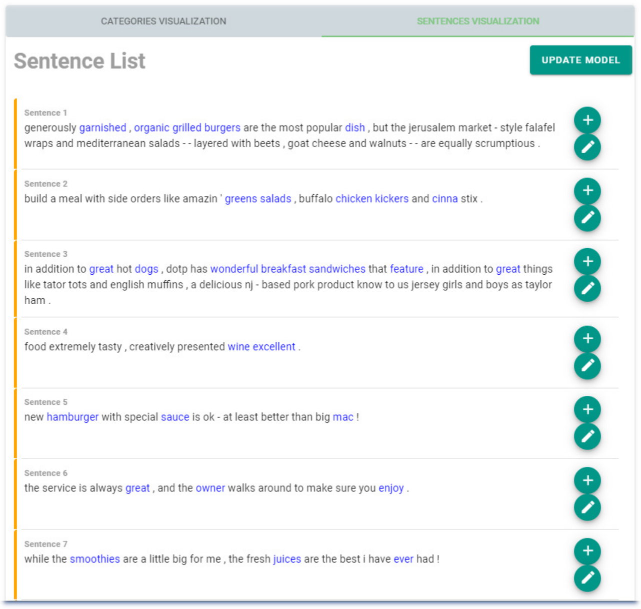
The main page provides a list of pre-trained models. New compatible models trained with ABAE can be added just placing them in a specific folder. Once the model is selected, the user can edit and validate at two levels: i) aspect inventory; and ii) given a dataset, at the context-aware aspect term level.
Figure 2 shows how the aspect inventory looks. It allows to rename, order and delete aspects. The aspects are organized using a hierarchical tree. This decision is motivated due to the observation that related aspects tend to fall into the same branches. We used a hierarchical agglomerative clustering algorithm to calculate the tree (Johnson, 1967). It was applied on the aspect representations from the aspect matrix T of the model. The UI also shows some representative terms for each aspect. These terms are calculated as the nearest word embeddings with respect to the aspect representations of T; based on cosine similarity.
Aspect On provides an interface for conducting and editing context-aware aspect term annotations. See Figure 3. We consider this option important for the user as it allows to verify and edit aspects while observes how they are annotated in specific contexts. It first requires the user to upload a dataset so ABAE can conduct the automatic annotation. Next, as can be seen in Figure 3a, the user has the option to use the edit sentence button and modify which words are annotated as aspect terms (Figure 3b). The *Update model* button allows to use the sentences revised and edited by the user to fine-tune the ABAE model weights; as described in the online learning process of Section 3.2..
Finally, Aspect On includes options to load and export the current aspect inventory and the updated ABAE model.

4. Evaluation

In this section we evaluate and compare the performance of Aspect On in two different domains. We also study the impact of its online learning method for aspect extraction.

4.1. Datasets and Methodology

Datasets We evaluate Aspect On with the Semeval 2014 Task 4 dataset on aspect based sentiment analysis (Pontiki et al., 2014). It contains two English domain-specific partitions, laptop and restaurant customer reviews, with aspect



Sentence List

Sentence 1
generously garnished , organic grilled burgers are the most popular dish , but the jerusalem market - style falafel wraps and mediterranean salads - - layered with beets , goat cheese and walnuts - - are equally scrumptious .

Sentence 2
build a meal with side orders like amazin ' greens salads , buffalo chicken kickers and cinna stix .

Sentence 3
in addition to great hot dogs , dotp has wonderful breakfast sandwiches that feature , in addition to great things like tator tots and english muffins , a delicious nj - based pork product know to us jersey girls and boys as taylor ham .

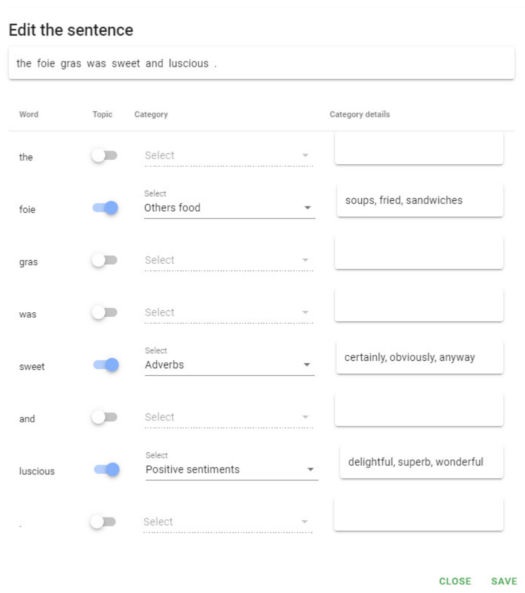
Sentence 4
food extremely tasty , creatively presented wine excellent .

Sentence 5
new hamburger with special sauce is ok - at least better than big mac !

Sentence 6
the service is always great , and the owner walks around to make sure you enjoy .

Sentence 7
while the smoothies are a little big for me , the fresh juices are the best i have ever had !

(a) UI for the sentences annotated with the aspect terms.



Edit the sentence

the foie gras was sweet and luscious .

Word	Topic	Category	Category details
the		Select	
foie		Others food	soups, fried, sandwiches
gras		Select	
was		Select	
sweet		Adverbs	certainly, obviously, anyway
and		Select	
luscious		Positive sentiments	delightful, superb, wonderful
.		Select	

(b) UI to edit the aspect terms of a sentence.

Figure 3: Context-aware aspect term annotation.

Domain	Sentences	Aspect terms
Laptops	3,845	917
Restaurants	2,717	1168

Table 1: Semeval 2014 Task 4 dataset statistics.

term annotations at the sentence level. The objective is to measure the quality of our system at detecting those aspect terms. The statistics of the partitions are summarized in Table 1.

Methodology We compare the Aspect On performance with the ABAE one when no online learning is used. Note that we use the modified version described in Section 3.1.2.. In addition, we show the Aspect On results with three optimizers that differ notably in performance when the annotations from the online learning method are employed: Adam (Kingma and Ba, 2014), Adagrad (Duchi et al., 2011), and Stochastic Gradient Descent (SGD) (Robbins and Monro, 1951).

We compare the Precision (P), Recall (R), and F-measure (F_1) of the systems. To measure the impact of the online learning, we also include a metric that determines the amount of effort that the user needs to revise and edit the automatically annotated data: the sum of the Mouse-Action Ratio (MAR) (Barrachina et al., 2009) of all the sentences. This metric has been widely used in statistical machine translation. It measures the ratio between the number of mouse actions required by the user to achieve the final reference (revised) sentence and its total number of words.

4.2. Technical Details and Parameters

We follow He et al. (2017) for most of the ABAE-related details. The ABAE and Aspect On base models[3] are trained using Adam with a learning rate of 0.001, 15 epochs, and a batch size of 50. The Aspect On online learning method uses that same learning rate and epochs regardless the optimizer. We initialize the matrix of aspects T based on K-means cluster centroids. We use it to cluster the word embbeddings of the dataset vocabulary. We use the top 20,000 most frequent words, after removing the stopwords, as the vocabulary of each dataset. We use fastText (Bojanowski et al., 2017) to obtain 300-dimensional word embeddings on the English Wikipedia. These embeddings remain frozen during the training to ease the network sentence reconstruction. We use K =20 aspects (T matrix rows) in all our experiments. Higher number of aspects did not provide with better results.
We measure the impact of the online learning by simulating user annotations (post-edits). These annotations are actually the gold standard annotations of the aspect terms. Therefore, given the ABAE dataset annotations, we use the gold standard to "post-edit" a certain number of them. Next, we run the Aspect On online learning with that post-edited portion of data. Finally, we measure the quality of that updated model when annotating the whole dataset. This evaluation procedure has been frequently used to save resources (Ortiz-Martínez et al., 2010; Domingo et al., 2019).

4.3. Results and discussion

We first compare the quality of the systems and the effort needed to post-edit their results. We show the results in

[3]All the models are implemented using `Tensorflow` (Abadi et al., 2016). We adapted the following ABAE implementation: `https://github.com/harpaj/Unsupervised-Aspect-Extraction`

	Restaurant reviews				Laptop reviews			
System	P [↑]	R [↑]	F_1 [↑]	MAR [↓]	P [↑]	R [↑]	F_1 [↑]	MAR [↓]
ABAE	0.57	0.56	0.52	0.55	0.57	0.57	0.58	0.39
Aspect On (Adam)	0.61	0.60	0.60	0.50	0.63	0.63	0.64	0.29
Aspect On (SGD)	**0.65**	**0.64**	**0.64**	**0.45**	**0.70**	**0.70**	**0.69**	**0.26**
Aspect On (Adagrad)	0.59	0.60	0.55	0.52	0.59	0.58	0.59	0.35

Table 2: General results with the Semeval 2014 Task 4 dataset.

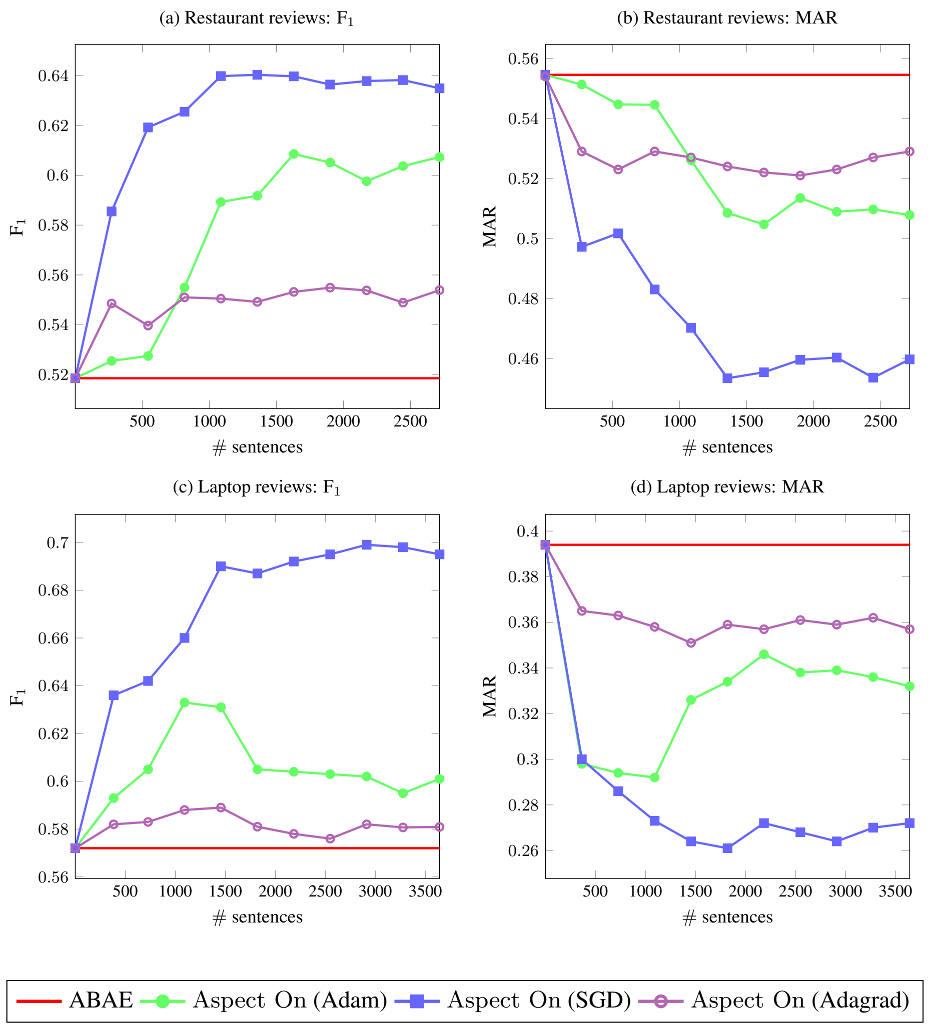


Figure 4: F_1 and MAR in function of the number of post-edited sentences.

Table 2.[4] Thanks to the online learning process, Aspect On outperforms ABAE in all the measures. In addition, we can see that the MAR is notably reduced when we use our model. Therefore, the effort needed to post-edit the model annotations is also reduced. The comparison of the Aspect On optimizers shows that SGD is the most adequate one to fine-tune the model. This might be produced by the Adam and Adagrad tendency to not converge to the optimal solution in certain tasks and datasets (Wilson et al., 2017). However, this could change with a deeper hyperparameter selection, which is out of the scope of this work. We should note that while conducting experiments we observed that, after a certain number of post-edited sentences, Aspect On converges and its performance and derived effort saving remains stable. Next, we investigate that point further.

We show the evolution of the F_1 and MAR in function of the post-edited number of sentences in Figure 4. Both dataset partitions show a similar trend at different scales depending on the Aspect On optimizer. As we can see, Adagrad obtains the smallest improvements compared to the ABAE baseline. In contrast, the SGD optimizer offers the best results and excels notably in both metrics. Focusing on that system, 500 sentences are enough to get an important boost in performance and reduction of effort. The peak occurs after providing the system with 1,000 and 1,500 sentences for the restaurant and laptop partitions, respectively.

[4]In this work, statistically significant results according to a χ^2 test are highlighted in bold.

That represents around 40% of the partition sentences. After that point, the improvements are limited or nonexistent. This manifests the capability and limitations of Aspect On to adapt to the user feedback. However, as we previously mentioned, these limitations could change with a deeper hyperparameter selection, which is out of the scope of this work.

5. Conclusions

In this work we introduced Aspect On, an interactive solution based on online learning that allows users to post-edit and improve the quality of an automatic aspect extraction. We believe a solution that is based on a solid backend and aware of the amount of effort needed to align aspects with the business need to be both wanting in the industry today. The presented approach is also very novel in its way of using user feedback to tune two separate parts of the neural model to reduce the manual work while increasing accuracy.

To the best of our knowledge, existing solutions in the industry either use a pre-determined list of topic words or provide a platform that requires customers to list the aspect words. In both cases, a constant update of the lexicons is necessary which forms a serious bottleneck in the federate analytics engines. That is because, in both cases, the solutions fail to exploit the richness, in coverage and in word relatedness, of the word embeddings while allowing the user to operate at the lexical level, i.e. editing a list of topics, to customize the engine. We believe that Aspect On, brings a significant novelty in this domain as it seamlessly links the symbolic aspects of the text (lexical and distributional) and allows the user to tune a neural model by simply giving feedback on the annotated results observed on few sentences and an elegantly organize tree describing the aspect as categories.

Our experimental results with the Semeval 2014 Task 4 dataset showed that Aspect On dramatically reduces the user effort required to post-edit the aspect extraction. The results also showed that our online learning algorithm can use the user edits to improve the aspect extraction of our model.

In the future, we plan to expand the capalities of Aspect On by allowing the user to transfer knowledge about aspects from different projects to further descrease the amount of manual work and increase the accuracy and efficiency.

Acknowledgments

We would like to thank their contributions to Angelo Basile who helped us improve the quality of this paper.

References

Abadi, M., Barham, P., Chen, J., Chen, Z., Davis, A., Dean, J., Devin, M., Ghemawat, S., Irving, G., Isard, M., et al. (2016). Tensorflow: A system for large-scale machine learning. In *12th {USENIX} Symposium on Operating Systems Design and Implementation ({OSDI} 16)*, pages 265–283.

Anderson, T. (2008). *The theory and practice of online learning*. Athabasca University Press.

Barrachina, S., Bender, O., Casacuberta, F., Civera, J., Cubel, E., Khadivi, S., Lagarda, A., Ney, H., Tomás, J., Vidal, E., et al. (2009). Statistical approaches to computer-assisted translation. *Computational Linguistics*, 35(1):3–28.

Blei, D. M., Ng, A. Y., and Jordan, M. I. (2003). Latent dirichlet allocation. *Journal of machine Learning research*, 3(Jan):993–1022.

Bojanowski, P., Grave, E., Joulin, A., and Mikolov, T. (2017). Enriching word vectors with subword information. *Transactions of the Association for Computational Linguistics*, 5:135–146.

Brody, S. and Elhadad, N. (2010). An unsupervised aspect-sentiment model for online reviews. In *Human Language Technologies: The 2010 Annual Conference of the North American Chapter of the Association for Computational Linguistics*, pages 804–812. Association for Computational Linguistics.

Chechik, G., Sharma, V., Shalit, U., and Bengio, S. (2010). Large scale online learning of image similarity through ranking. *Journal of Machine Learning Research*, 11(Mar):1109–1135.

Denkowski, M., Dyer, C., and Lavie, A. (2014). Learning from post-editing: Online model adaptation for statistical machine translation. In *Proceedings of the 14th Conference of the European Chapter of the Association for Computational Linguistics*, pages 395–404.

Domingo, M., García-Martínez, M., Peris, Á., Helle, A., Estela, A., Bié, L., Casacuberta, F., and Herranz, M. (2019). Incremental adaptation of nmt for professional post-editors: A user study. *arXiv preprint arXiv:1906.08996*.

Duchi, J., Hazan, E., and Singer, Y. (2011). Adaptive subgradient methods for online learning and stochastic optimization. *Journal of Machine Learning Research*, 12:2121–2159.

Ganu, G., Elhadad, N., and Marian, A. (2009). Beyond the stars: improving rating predictions using review text content. In *WebDB*, volume 9, pages 1–6. Citeseer.

He, R., Lee, W. S., Ng, H. T., and Dahlmeier, D. (2017). An unsupervised neural attention model for aspect extraction. In *Proceedings of the 55th Annual Meeting of the Association for Computational Linguistics*, pages 388–397.

Hu, M. and Liu, B. (2004). Mining and summarizing customer reviews. In *Proceedings of the tenth ACM SIGKDD international conference on Knowledge discovery and data mining*, pages 168–177. ACM.

Johnson, S. C. (1967). Hierarchical clustering schemes. *Psychometrika*, 32(3):241–254.

Karamanolakis, G., Hsu, D., and Gravano, L. (2019). Leveraging just a few keywords for fine-grained aspect detection through weakly supervised co-training. *arXiv preprint arXiv:1909.00415*.

Kingma, D. P. and Ba, J. (2014). Adam: A method for stochastic optimization. *arXiv preprint arXiv:1412.6980*.

Li, G., Hoi, S. C., Chang, K., and Jain, R. (2010). Micro-blogging sentiment detection by collaborative online learning. In *2010 IEEE International Conference on Data Mining*, pages 893–898. IEEE.

Liu, B. (2012). Sentiment analysis and opinion mining. *Synthesis lectures on human language technologies*, 5(1):1–167.

MacQueen, J. et al. (1967). Some methods for classification and analysis of multivariate observations. In *Proceedings of the fifth Berkeley symposium on mathematical statistics and probability*, volume 1, pages 281–297. Oakland, CA, USA.

McAuley, J., Leskovec, J., and Jurafsky, D. (2012). Learning attitudes and attributes from multi-aspect reviews. In *2012 IEEE 12th International Conference on Data Mining*, pages 1020–1025. IEEE.

Mukherjee, A. and Liu, B. (2012). Aspect extraction through semi-supervised modeling. In *Proceedings of the 50th annual meeting of the association for computational linguistics: Long papers-volume 1*, pages 339–348. Association for Computational Linguistics.

Ortiz-Martínez, D., García-Varea, I., and Casacuberta, F. (2010). Online learning for interactive statistical machine translation. In *Human Language Technologies: The 2010 Annual Conference of the North American Chapter of the Association for Computational Linguistics*, pages 546–554. Association for Computational Linguistics.

Pontiki, M., Galanis, D., Pavlopoulos, J., Papageorgiou, H., Androutsopoulos, I., and Manandhar, S. (2014). SemEval-2014 task 4: Aspect based sentiment analysis. In *Proceedings of the International Workshop on Semantic Evaluation (SemEval 2014)*, pages 27–35.

Robbins, H. and Monro, S. (1951). A stochastic approximation method. *The annals of mathematical statistics*, pages 400–407.

Titov, I. and McDonald, R. (2008). Modeling online reviews with multi-grain topic models. In *Proceedings of the 17th international conference on World Wide Web*, pages 111–120. ACM.

Wang, L. and Soong, F. K.-P. (2009). Handwriting-based user interface for correction of speech recognition errors. US Patent App. 12/042,344.

Wang, L., Liu, K., Cao, Z., Zhao, J., and De Melo, G. (2015). Sentiment-aspect extraction based on restricted boltzmann machines. In *Proceedings of the 53rd Annual Meeting of the Association for Computational Linguistics and the 7th International Joint Conference on Natural Language Processing (Volume 1: Long Papers)*, pages 616–625.

Wilson, A. C., Roelofs, R., Stern, M., Srebro, N., and Recht, B. (2017). The marginal value of adaptive gradient methods in machine learning. In *Advances in Neural Information Processing Systems*, pages 4148–4158.

Zhao, W. X., Jiang, J., Yan, H., and Li, X. (2010). Jointly modeling aspects and opinions with a maxent-lda hybrid. In *Proceedings of the 2010 conference on empirical methods in natural language processing*, pages 56–65. Association for Computational Linguistics.