Conceptual Similarity for Subjective Tags
Yacine Gaci, Boualem Benatallah, Fabio Casati, Khalid Benabdeslem

To cite this version:
Yacine Gaci, Boualem Benatallah, Fabio Casati, Khalid Benabdeslem. Conceptual Similarity for Subjective Tags. AACL-IJCNLP 2022 (2nd Conference of the Asia-Pacific Chapter of the Association for Computational Linguistics and the 12th International Joint Conference on Natural Language Processing), Nov 2022, Online, Taiwan. hal-03866253

HAL Id: hal-03866253
https://hal.science/hal-03866253
Submitted on 22 Nov 2022

HAL is a multi-disciplinary open access archive for the deposit and dissemination of scientific research documents, whether they are published or not. The documents may come from teaching and research institutions in France or abroad, or from public or private research centers.

L’archive ouverte pluridisciplinaire HAL, est destinée au dépôt et à la diffusion de documents scientifiques de niveau recherche, publiés ou non, émanant des établissements d’enseignement et de recherche français ou étrangers, des laboratoires publics ou privés.
Conceptual Similarity for Subjective Tags

Yacine Gaci  
LIRIS - University of Lyon 1, France  
yacine.gaci@univ-lyon1.fr

Boualem Benatallah  
Dublin City University, Ireland  
University of New South Wales, Australia  
boualem.benatallah@gmail.com

Fabio Casati  
University of Trento, Italy  
fabio.casati@gmail.com

khalid Benabdeslem  
LIRIS - University of Lyon 1, France  
khalid.benabdeslem@univ-lyon1.fr

Abstract

Tagging in the context of online resources is a fundamental addition to search systems. Tags assist with the indexing, management, and retrieval of online products and services to answer complex user queries. Traditional methods of matching user queries with tags either rely on cosine similarity, or employ semantic similarity models that fail to recognize conceptual connections between tags, e.g. ambiance and music. In this work, we focus on subjective tags which characterize subjective aspects of a product or service. We propose conceptual similarity to leverage conceptual awareness when assessing similarity between tags. We also provide a simple cost-effective pipeline to automatically generate data in order to train the conceptual similarity model. We show that our pipeline generates high-quality datasets, and evaluate the similarity model both systematically and on a downstream application. Experiments show that conceptual similarity outperforms existing work when using subjective tags.

1 Introduction

As products and services proliferated the Internet in recent years, tagging came into prominence to facilitate the consumption of online information (Smith, 2007). Tagging is the practice of assigning labels and keywords to online resources. It plays a pivotal role in the indexing, management and retrieval of factual information. On the other hand, recent years have witnessed a major shift in people’s expectations when searching online (Li et al., 2019). Beside the factual data such as a restaurant’s cuisine type or a camera’s resolution, the search trend evolved to be more experiential (Li et al., 2019). Common search queries include attributes such as delicious food for restaurants or long-lasting battery for cameras. Previous work (Li et al., 2019; Gaci et al., 2021) called this new set of attributes as subjective tags because they are short phrases that hint towards the subjective quality of products and services.

Subjective tags are particularly useful in enhancing online experiential search. In this context, users who are seeking subjective experiences, include sets of tags they care about in their queries, and it is the search system’s responsibility to fetch products and/or services that have been previously described with matching tags. Deciding whether two given subjective tags match or not implies using a similarity measure, for which cosine similarity remains a convenient, yet arbitrary default (Zhelezniak et al., 2019; Li et al., 2019; Chang et al., 2019). Recent search systems such as OpineDB (Li et al., 2019) or SearchLens (Chang et al., 2019) rely mostly on cosine similarity when it comes to comparing tag-like short phrases, since it is easy to use and provides simple geometric interpretations (Zhelezniak et al., 2019). However, recent studies (May et al., 2019; Zhou et al., 2022) argue that this interpretability becomes fogged when dealing with sentences or phrases, and cosine similarity suffers from severe limitations when used to compare multi-word textual inputs.

A lot of research has been directed toward proposing supervised methods for textual similarity, spanning a diverse set of paradigms, e.g. Siamese networks (Bromley et al., 1993; Ranasinghe et al., 2019), Aggregation-Matching models (Wang and Jiang, 2016; Wang et al., 2016, 2017), or the recent cross-sentence attention paradigm which was made possible by the advent of the transformer architecture (Vaswani et al., 2017). Although these models work fairly well on syntactically-correct sentences (Bethard et al., 2017), they lack effectiveness when used with shorter-spanned phrases such as subjective tags. A reason behind this is that subjective tags do not share the same structure of full sentences and hence require different treatment. As will be discussed later in this paper, our experiments confirm this limitation. A
second drawback is that current similarity models are not explicitly trained to recognize conceptual similarities between the compared textual entities (e.g., meal and pizza share the concept of food; or background music and lighting share the concept of ambiance). Therefore, all conceptual reasoning is disregarded. In this work, we compel our own similarity model to encode more conceptual relationships as provided by a human (whom we call the designer) and further expanded by popular knowledge bases such as WordNet (Fellbaum, 2012) or ConceptNet (Speer et al., 2017).

To illustrate the importance of capturing conceptual similarities between subjective tags, suppose a user searches for a restaurant serving delicious meals. A search system should be able to suggest a restaurant which has been tagged with tasty chicken wings among its search results, because meal and chicken wings share the same concept (that of food) even though meal and chicken wings are not semantically similar. As a result, traditional semantic similarity models (Bethard et al., 2017; Li et al., 2019; Ranasinghe et al., 2019) usually fail to meet this expectation and provide low similarity scores for the tags in the example. The same reasoning applies to other subjective tags, like high-autonomy camera and long-lasting battery, or romantic ambiance and low-beat music bar.

Aiming to solve the aforementioned drawbacks, we propose a new similarity model that focuses on learning and then using conceptual relationships as reflected in the training data. Given the new nature of subjective tags (Li et al., 2019; Gaci et al., 2021), we are not aware of the existence of datasets that suit our needs. Besides, manually annotating data is expensive, and extending to other application domains (e.g. from restaurants to electronics) usually necessitates re-annotating from scratch. Therefore, the main contribution of this paper is a pipeline to automatically generate large synthetic datasets for the conceptual similarity task. First, we prompt the dataset designer to provide seed words for the concepts she needs her conceptual similarity model to learn about. Second, we exploit the simple structure of subjective tags (Gaci et al., 2021) to expand the seeds with conceptually related terms using knowledge bases, or the implicit knowledge encoded in existing language models to automatically generate large training data.

Our second contribution is the similarity model itself. Capitalizing on the latest advances in semantic similarity research (Ranasinghe et al., 2019; Wang et al., 2017; Devlin et al., 2018), we propose a new similarity model by combining insights from aggregation-matching and cross-sentence attention paradigms. We show that conceptual similarity is better than cosine similarity with a margin of 17.42% in terms of Pearson correlation, or BERT-based similarity models through systematic evaluations. We also plug different similarity models into a tag-based search system and show that conceptual similarity outperforms them all. Also, we evaluate the quality of the automatically generated dataset through various experiments. We release our code and data in GitHub 1.

2 Related Work

2.1 Synthetic Dataset Generation

Acquiring training data is increasingly the largest and most pressing bottleneck in deploying machine learning systems (Ratner et al., 2017). The traditional way of doing so calls a team of experts to manually create and then label the data, incurring enormous costs. Crowdsourcing alleviates part of this burden by proposing to a group of individuals of varying knowledge and expertise, the undertaking of the labeling task (Brabham, 2013; Howe, 2006). However, crowdsourcing runs the risk of corrupting the precision of the gold labels, and may inflict noise in the labeling process, especially when uneducated, careless or malicious workers are involved. A recent trend for acquiring training data is devising methods to automatically create, generate and label these critical building blocks of supervised learning systems with little effort (Ratner et al., 2016, 2017; Varma and Ré, 2018). When one speaks of generating data, two problems are implicitly addressed: (1) generation of features (i.e. unlabeled raw data), and/or (2) generation of gold labels (i.e. automatic labeling).

First, we discuss the generation of features, for which two techniques are mainly used: template-based generation (Dev et al., 2020; Nadeem et al., 2020; Ribeiro et al., 2020) and data augmentation (Zhao et al., 2018; Zmigrod et al., 2019; Taylor and Nitschke, 2017; Nie et al., 2020; Kaushik et al., 2019). In template-based generation, a set of tokens iteratively replaces the placeholders in templates, creating a separate example each time. Dev et al. (2020) provide templates such as "The [PLACE-
weak classifiers including accuracy, coverage, and
we propose five different expansion techniques to
without changing the labels (Kaushik et al., 2019),
where users construct their own test benchmarks
HOLDER] is a doctor”, and insert words like man,
woman, muslim, christian to create different ex-
amples to study social biases and stereotypes. In
the same spirit, Nadeem et al. (2020) construct an
evaluation dataset of biases through the use of tem-
plates and crowdsourcing, whereas Ribeiro et al.
(2020) designed a framework to test NLP systems
where users construct their own test benchmarks
via the use of templates. On the other hand, data
augmentation techniques expect an already avail-
able set of data, that they augment and expand
to create larger sets. This is usually achieved by
searching for similar inputs in the feature space,
applying small perturbations to the existing data
without changing the labels (Kaushik et al., 2019),
or through seed expansion techniques (Fast et al.,
2016; Li et al., 2019; Huang et al., 2020) via similarity
in word embeddings or with knowledge bases.

Our own data generation is a mix of both tech-
niques. While it is fundamentally a seed expansion
method where aspect and opinion terms that we
use to express subjective tags are expanded into
conceptually related terms, it also derives from
template-based generation since we use the tem-
plate "<opinion> <aspect>" (as in delicious food
or romantic ambiance) to construct subjective tags.
The closest work to ours in terms of seed expansion
is Empath (Fast et al., 2016) for studying topic sig-
als in text. In Empath, a topic is defined by a set of
seeds that are later expanded by either using word
embeddings or crowdsourcing, to enrich each topic
category. In contrast, we use the expansions to
build sufficiently large labeled datasets. Moreover,
we propose five different expansion techniques to
increase the diversity of generated subjective tags.

The second problem in automatic data genera-
tion is generating the ground truth labels. Data pro-
gramming (Ratner et al., 2016) is a recent paradigm
that enables the programmatic creation of large-
scale training sets in which different weak super-
vision sources (e.g. heuristics, knowledge bases,
crowdsourcing) are combined. In Snorkel (Rat-
ner et al., 2017) and Snuba (Varma and Ré, 2018),
combination is done with a generative model that
takes into consideration several properties of the
weak classifiers including accuracy, coverage, and
inter-correlations. Our work is different in two
main aspects. First, Snorkel and Snuba are general
frameworks that present general guidelines aiming
to build labeling functions, whereas our method is
much more specific, and focuses on similarity for
subjective tags. Second, in this work, we generate
and label training sets at the same time, in con-
trast to Snorkel whose purpose is to assign labels
to already existing unlabeled data.

2.2 Textual Similarity
Apart from cosine similarity, we identify several
similarity paradigms in the literature: (1) Siamese
networks (Bromley et al., 1993; Ranasinghe et al.,
2019) where the same encoder is used to project
inputs into the same embedding space. Then, the
similarity decision is made based on the vector
representations alone. (2) Aggregation-matching
paradigm (Wang and Jiang, 2016; Wang et al.,
2016, 2017) which adds explicit matchings be-
tween the representations of inputs, before aggre-
gating them and computing similarity. (3) Cross-
sentence attention paradigm which is enabled by
finetuning transformer models such as BERT on a
similarity task (Devlin et al., 2018; Peinelt et al.,
2020). (4) Combining several weak similarity mod-
els such as simple neural networks, tree-based
and/or probabilistic models through an ensemble
(Bethard et al., 2017; Tian et al., 2017; Lair et al.,
2020). However, all these works focused solely on
semantic similarity between syntactically correct
sentences, whereas we focus on conceptual sim-
ilarity between tag-like short phrases, similar to
Anuar et al. (2015); Zhu and Iglesias (2016). In
contrast, we use knowledge graphs to generate data
and train a supervised model. More details about
our similarity model are provided in Section 4.

3 Pipeline to Generate Training Datasets
Borrowing from the Aspect-Based Sentiment An-
alysis literature (Liu, 2012; Gaci et al., 2021), we
define a subjective tag as the concatenation of an
aspect term with an opinion term. The aspect term
designates the component or the feature being de-
scribed and the opinion term characterizes this fea-
ture. For example, delicious food is a subjective
tag wherein food is the aspect while delicious is the
opinion. This definition is sufficiently expressive
to allow a wide range of subjective tags such as ro-
matic ambiance, clean hotel rooms, long-lasting
battery, great camera or amiable dentist.

Specific to this work, we define a concept as
a set of aspect terms conceptually related to each
other. For example, the concept of food can be
described with the following set of terms: {food,
plates, dishes, pizza, chicken wings, meal, pasta}
while the concept of ambiance can be defined with \{ambiance, atmosphere, lighting, background music, dance floor\}. The goal of conceptual similarity is to consider the aspects belonging to the same concept as similar when described with similar opinions.

We cast conceptual similarity as a binary classification problem, where the positive label denotes similarity. These specifications enable automatic generation of high-quality labeled datasets for conceptual similarity of subjective tags, with minimal costs. To do so, the dataset designer provides a list of concepts. We then leverage seed expansion techniques to generate the dataset, through the pipeline illustrated in Figure 1. In the following, we describe each step of the pipeline in detail.

### 3.1 Providing Concept Seed Words

The first step in the pipeline is to provide seed words for the concepts that the dataset designer wants to take into consideration. For each concept \( i \), the designer provides a list of aspect seed words \( A_i \), and \( m_i \) lists of opinion seed words \( O_{ij} \) where \( j \in \{1...m_i\} \); \( m_i \) depends on the concept and the level of granularity the dataset designer aims to reach. For the sake of illustration, say that the designer wants to include the concept of food with three classes of opinions (delicious, horrible, healthy). She may provide the following:

\[
A_i = \{ \text{“food”, “dish”, “lunch”, “pizza”, “snack”} \} \\
O_{ij}^1 = \{ \text{“good”, “delicious”, “excellent”} \} \\
O_{ij}^2 = \{ \text{“bad”, “horrible”, “not seasoned”} \} \\
O_{ij}^3 = \{ \text{“healthy”, “organic”, “high quality”} \}
\]

\( A_i \) lists aspect terms related to the concept of food. Each of \( O_{ij} \) lists some opinion terms of the same nature, but different from one set to another. In the example above, \( O_{ij}^1 \) describes tasty food, \( O_{ij}^2 \) characterizes bad food, and \( O_{ij}^3 \) deals with healthy food. In this particular scenario, conceptual similarity trained on a dataset to be generated from these seed words considers the tags “good food” and “healthy food” as dissimilar because the terms good and healthy belong to different opinion sets.

If the dataset designer needs a more granular similarity model (like spicy food described as its own class), she only has to add another set with seed words depicting spiciness. Following these guidelines, the designer can express a wide range of concepts such as price, service, hygiene, and in other domains too (hotels, electronics, books, etc.).

### 3.2 Seed Word Expansion

We propose five different techniques to expand the set of seed words given by the dataset designer. We illustrate these techniques in Figure 2 and describe them in the following:

**WordNet Expansion.** For every seed, we collect its corresponding synsets from WordNet (Fellbaum, 2012). Then, for every synset, we retrieve its hyponyms, hypernyms, meronyms and sister terms as illustrated in Figure 2(a). We control the number of expansions through the use of hyperparameters such as the maximum number of synsets to include, and different booleans each telling whether we take hyponyms, meronyms, etc. respectively.

**ConceptNet Expansion.** For every seed, we obtain its is-a (i.e. parent concepts) and type-of (child concepts) relations. For example, meat and food are parent concepts for the word of interest, i.e. chicken. We also retrieve other children of the parent concepts as is shown in Figure 2(b). We control ConceptNet expansion with three hyperparameters: capacity which is the maximum number of relations to consider; minimum weight which specifies the relevance of the relation (high weights in ConceptNet (Speer et al., 2017) correspond to a strong relation); and a boolean specifying whether to include children of parent concepts into the expansion.

**Word Embedding Expansion.** The goal is to find the \( top_k \) words in the vocabulary that minimize the total distance between them and seed terms. Taking the example in Figure 2(c), pasta is less distant from all the seeds than morning is, thus pasta constitutes a better expansion. The parameters of this technique is the number of expansions.
top_k, the word embedding model under use, and the distance function, e.g. euclidean.

**Language Generation Expansion.** This method plugs seed words into a template such as "These concepts are related: <seed_1>, <seed_2>, ... <seed_n>, and ", then asks an autoregressive language model to generate a continuation for this sentence. We then take the top_k words having the highest probabilities to be correct continuations. The hyperparameters are: the language model (e.g. GPT2, T5), the number of generations, and the maximum length of each generated expansion.

**Masked Language Modeling Expansion.** Similar to the previous expansion technique, we use a masked language model (Devlin et al., 2018), where the template takes the following form: "<seed_1>, <seed_2>, ... <seed_n> and [MASK] are all related concepts." The masked language model produces, for every word in the vocabulary, its likelihood to replace the mask. So terms having the same concept as the seeds have higher probabilities. The parameters of this method are the number of top_k terms to take, and the masked language model under use, e.g. BERT, Albert...

For every expansion technique, we can have as many expanders as there are parameter configurations. For example, two word embedding expanders, one based on Word2vec while the other on GloVe, are two different expanders. Or one that uses an euclidean distance while the other uses cosine similarity are also different expanders. We give the full list of parameter configurations we used for every expansion method in our experiments in Section A.2. For a new word to be considered as a correct expansion, we require that at least a sufficient number of expanders suggest the word. We specify this with \textit{min\_consensus\_rate} which defines how many expanders need to produce the word in order to include it in the final expansions.

### 3.3 Random Sampling

We randomly choose an aspect term from one of the expanded aspect sets, and an opinion term from one of its associated opinion sets. These two terms are concatenated to form a subjective tag. For example, we may sample the aspect term waiters and the opinion term nice to form the tag "nice waiters". We repeat this process to construct as many subjective tags as the dataset designer needs.

### 3.4 Filtering

Random sampling from automatically generated sets of terms may lead to arbitrary tags. For instance, it may construct tags such as "helpful duty".\footnote{This may be the result of expanding service to duty through WordNet, even though service in this case refers to the waiters in a restaurant} We eliminate those tags by using a language model which assigns likelihoods to sentences.
sentences so that semantically sound sentences are given high likelihoods and low quality sentences get low likelihoods. We use GPT2 language model (Radford et al., 2019) by feeding it with subjective tags formatted according to this template: "the aspect is opinion". GPT2 should assign low probabilities to sentences such as "the duty is helpful", and high probabilities to sentences such as "the service is helpful" or "the waitstaff is agreeable".

We manually select the probability threshold above which sentences make sense, and keep the generated tags that score above that threshold.

3.5 Pairing and Labeling

We randomly sample two subjective tags $t_1$ and $t_2$ from the filtered list. If the aspect and opinion terms of $t_1$ and $t_2$ have been sampled from the same sets, the tags are considered similar (label is 1). In all other cases, the label is 0. To avoid class imbalance in the dataset, the dataset designer provides the minimal ratio of positive examples. We enforce this constraint by deliberately sampling similar tags from the same aspect and opinion sets.

Figure 1 summarizes our dataset generation pipeline with an example. This algorithm allows us to create high-quality training datasets with minimal effort. It can also be adapted to any domain. In Section 5.2, we evaluate the quality of datasets generated with this pipeline.

4 Conceptual Similarity Model

In this section, we present our approach to compute conceptual similarity for a pair of subjective tags. Following guidelines from the aggregation-matching paradigm (Wang and Jiang, 2016), our model encodes explicit interactions between tags, e.g. whether the tags correspond to the same concept; whether they use the same opinions but with different aspects; whether the choice of words in the tags is similar but the tags themselves are not. To this end, we propose a novel bilateral matching model which automatically encodes such interactions and relationships before making a similarity decision. Given two subjective tags $t_1$ and $t_2$, this model estimates their similarity by computing their probability of being perfectly similar $P(sim = 1|t_1, t_2)$. Figure 3 illustrates the different layers of this model.

We begin by feeding $t_1$ and $t_2$ into BERT (Devlin et al., 2018). This serves two purposes: First, we get word embeddings for each word in the tags; second, we have a CLS vector that captures the relationship between $t_1$ and $t_2$ as a vector. Given BERT embeddings $[u_1, ..., u_m]$ and $[v_1, ..., v_n]$, we utilize mean pooling to obtain fixed-sized embeddings for each tag ($u^all$ and $v^all$). The next layer in the network matches each word embedding of one tag with all the word embeddings of the other tag. The matching is done in two directions (hence the bilateral aspect): (1) We match each $u_i$ with $v^all$ to compare each word $u_i$ in $t_1$ with all the words in $t_2$, and encode their relationship. (2) We match each $v_i$ with $u^all$ to do the same in the reverse direction.

The matching function we use is the element-wise multiplication which has long been used in the NLP community as a proxy for similarity. Thus, we use it to match word embeddings of $t_1$ and $t_2$. After the matching layer, we aggregate $[u'_1, ..., u'_m]$ and $[v'_1, ..., v'_n]$ to obtain fixed-length vectors for each tag via Bidirectional LSTM (BiLSTM) layers (Hochreiter and Schmidhuber, 1997), taking the last hidden states as tag embeddings $u$ and $v$. At this step, we have encoded the relationship between $t_1$ and $t_2$ using two different paradigms: (1) aggregation-matching through the use of element-wise multiplication for matching and BiLSTM for aggregation (vectors $u$ and $v$), and (2) the cross-sentence attention paradigm through CLS vector, because BERT uses self-attention (Vaswani et al., 2017) to compute its vectors. We concatenate $u$, $v$ and CLS and feed it to a classification head (FFNN
layer) to estimate similarity.

5 Experiments

We use Restaurants as the test domain. We consider nine concepts that we use to automatically generate the training dataset: Food, Service, Price, Atmosphere, Location, Cleaning, Environment, Menu and Parking. Each concept consists of one set of aspect terms, and two to three sets of different opinion terms. The choice of concepts, and seed words for aspects and opinions was inspired by previous work (Moura et al., 2017) who conducted surveys and qualitative experiments on many restaurant-seeking participants, and identified the most important factors taken into account by these same participants in their decision-making process for choosing a restaurant. The full list of concepts and their seeds is in Section A.3, while the hyperparameter details for the similarity model are in Section A.1. In the following, we first compare conceptual similarity to various baselines. Next, we evaluate the quality of the automatically generated dataset. Finally, we assess the practical value of conceptual similarity by measuring its impact on a downstream search system proposed by Gaci et al. (2021) that uses subjective tags.

5.1 Evaluating Conceptual Similarity

Existing similarity benchmarks provide similarity ground truth for syntactically correct sentences (Bethard et al., 2017). Hence, we cannot use them given that subjective tags are short phrases which do not draw from the same syntactically-complete sentence distribution. To the best of our knowledge, no benchmark for subjective tags exists. For this reason, we create our own test set by automatically extracting tags from Yelp’s restaurant online reviews using the tag extractor of SACCS (Gaci et al., 2021). Given a snippet of text, SACCS extracts subjective tags as concatenations of aspects and opinions. We then map these extracted tags randomly into pairs. We select 500 such pairs and ask three participants to assign a similarity score between 0 and 5 for each pair of subjective tags. We then normalize the similarity scores to squash them into the unit range before taking the mean across the participants. As in standard similarity evaluations, we use three metrics: Pearson and Spearman correlation, and Mean Absolute Error (MAE).

<table>
<thead>
<tr>
<th>Similarity Model</th>
<th>Pearson</th>
<th>Spearman</th>
<th>MAE</th>
</tr>
</thead>
<tbody>
<tr>
<td>Cosine (Word2vec)</td>
<td>0.6770</td>
<td>0.6190</td>
<td>0.2083</td>
</tr>
<tr>
<td>Cosine (BERT MEAN)</td>
<td>0.3449</td>
<td>0.3312</td>
<td>0.5313</td>
</tr>
<tr>
<td>Cosine (BERT CLS)</td>
<td>0.0497</td>
<td>0.0848</td>
<td>0.6920</td>
</tr>
<tr>
<td>BERT Classif</td>
<td>0.5946</td>
<td>0.5404</td>
<td>0.1703</td>
</tr>
<tr>
<td>Random Forest</td>
<td>0.6271</td>
<td>0.6324</td>
<td>0.2614</td>
</tr>
<tr>
<td>Siamese</td>
<td>0.7058</td>
<td>0.6141</td>
<td>0.1903</td>
</tr>
<tr>
<td>Conceptual Sim</td>
<td>0.8512</td>
<td>0.7388</td>
<td>0.1134</td>
</tr>
</tbody>
</table>

Table 1: Evaluation of similarity models

We compare our conceptual similarity model to several baselines: A Siamese network (Ranasinghe et al., 2019) and a random forest classifier with hand-crafted features (Tian et al., 2017), both trained on the same dataset we use to train our own model. Also, owing to the universality of cosine similarity, we compare against it both with Paragraph embeddings (Wieting et al., 2015) and on BERT embeddings with different pooling methods, MEAN and CLS as in Devlin et al. (2018); Li et al. (2019). Finally, we train a BERT-based model that we augment with a classification head (BERT Classif) and finetune on the same training data we used to train our conceptual similarity to make it more competitive. Table 1 summarises the results.

We can see that conceptual similarity outperforms cosine similarity by a large margin (0.1742 points in Pearson correlation). This demonstrates that cosine should no longer be perceived as the default when it comes to measuring similarity for subjective tags. We also show that BERT alone cannot cater for a task as ambiguous as similarity for subjective tags, even when finetuned on the same training set that we use. This sheds light on the necessity to design custom models especially tailored for tag similarity. We argue that the effectiveness of our method stems from its ability to match different words of subjective tags using both attention and element-wise multiplication.

Existing information retrieval and tag-based search systems like Li et al. (2019) and Chang et al. (2019) blindly trust cosine similarity or a finetuned BERT without investigating their implications on the overall system performance. Our work highlights the limitations regarding main stream text similarity techniques for subjective tags and short phrases, as it gives guidelines as to how to design robust similarity models.

3https://www.yelp.com/dataset
5.2 Evaluating the Quality of Training Data

We measure the quality of the automatically generated training dataset by injecting artificial noise in the data and checking whether it degrades in quality (Jassar et al., 2009). We define noise in this context as swapping the labels in the training set. For example, if the original line in the dataset was \( t_1, t_2, 1 \), the new noisy line would be \( t_1, t_2, 0 \) and vice versa. We perturb fixed percentages of the training data (5%, 10%, 25% and 50%) and retrain the similarity model each time. The rationale of this experiment is that the introduction of noise should degrade the quality of training. In this spirit, if the similarity model trained on noisy data is of comparable accuracy to the one trained on the original unperturbed data, we argue that the original data was merely noise. On the other hand, if introducing noise degrades the performance of the similarity model, one can assume that the original data was of good quality. Table 2 shows the similarity correlations with human-defined scores as described in Section 5.1. We observe that instilling noise drops the accuracy of conceptual similarity. This reflects that the original unperturbed dataset is of high quality.

5.3 Experiments on a downstream System

In the following, we demonstrate the effectiveness of conceptual similarity when plugged into a downstream search application Gaci et al. (2021). We give a brief overview of the application, describe the baselines, benchmarks and evaluation metrics.

**System overview.** SACCS (Gaci et al., 2021) is a subjectivity-aware system to search for restaurants online. From their reviews, SACCS automatically extracts subjective attributes of restaurants offline in the form of subjective tags. Then, when users provide their search queries, they can include subjective tags as search filters. SACCS uses an underlying similarity model to compare between user-provided tags and those describing each restaurant. The final output of SACCS is a ranked list of restaurants ordered by relevance to the user query.

**Baselines.** We replace the similarity model used in SACCS with our conceptual similarity and the baselines we used in Section 5.1, to create as many baselines for this experiment.

**Evaluation benchmark.** We follow the same experiment used in Gaci et al. (2021) to assess the overall quality of the search system, and hence evaluate the practical value of conceptual similarity. Mainly, we use the same crowdsourced evaluation benchmark as in Gaci et al. (2021), consisting of subjective search queries with three levels of difficulty: Short queries have only one subjective tag; Medium queries have two; Long queries with three. Each difficulty level contains 100 different search queries, and each query is associated with a ranked list of relevant restaurants that best answer it.

**Evaluation metric.** We evaluate the final search quality using the popular Normalized Discounted Cumulative Gain (NDCG) (Christopher et al., 2008). The closer the score is to 1 using this metric, the better are the search results overall. Given that we use the same system in all the baselines of this experiment, and that these differ only in the underlying similarity model in use, we infer that the NDCG scores directly reflect the quality of the similarity models. Table 3 shows the results.

**Results.** Table 3 demonstrates the effectiveness of conceptual similarity, outperforming all other similarity models on all levels of difficulty, especially the universal cosine similarity which performs worse by a margin of 2.76%. This experiment proves that conceptual similarity is efficient when plugged in tag-based search applications.

6 Conclusion

In this work, we propose conceptual similarity for subjective tags. We also propose a methodology to
automatically generate training datasets for conceptual similarity with minimal effort given a domain and a set of concepts. Unlike traditional semantic similarity, our model is trained with conceptual signals as reflected in the generated dataset. Intrinsic and extrinsic experiments demonstrate the superiority of our approach on subjective tags.

On the other hand, we acknowledge the following limitations. Although the method is independent from the application domain, we constrained our evaluations to the Restaurants domain for reasons related to unavailability of test data. So we were forced to create our own test benchmark by asking three participants to give ground truth labels for 500 pairs of subjective tags. This may seem small-scale, which risks putting into question the conclusions regarding the superiority of our similarity approach. However, the extrinsic experiment that we conduct by using relatively larger crowdsourced data shows that our approach is efficient and outperforms other similarity models, which assuages our concern. As future work, we plan to apply our methods on other domains, e.g. hotels, or electronics.

In this paper, we build the whole argument of our contributions against the blind use of cosine similarity in tag-based search systems, and to replace it with our newly proposed conceptual similarity. However, we employ BERT and LSTMs in our model which incur a much higher computational cost than cosine similarity. The adoption of our model in practice depends on whether efficiency is a major concern in the downstream search application, i.e. whether a poor search inflicts major negative consequences in critical domains such as finances or regulations. It also depends on the underlying infrastructure into which conceptual similarity will be deployed, e.g. are there any GPUs in use? Is memory space enough to hold BERT and LSTMs? So whether to adopt our contributions in practice is a compromise between cost and efficiency.

References


We apply dropout with a ratio of 0.3. To train, we use a hidden dimension of 128 for the LSTM layer, and 512 for the 2-layer classification FFNN. We implemented conceptual similarity in Python using standard packages such as PyTorch\textsuperscript{4} for neural networks, HuggingFace transformers library\textsuperscript{5} for BERT and GPT2.

A Appendix

A.1 Similarity Model Details & Hyperparameters

We use a hidden dimension of 128 for the LSTM layer, and 512 for the 2-layer classification FFNN. We apply dropout with a ratio of 0.3. To train the model, we minimize cross entropy of the training set, and use Adam optimizer (Kingma and Ba, 2014) to update the parameters with $5e^{-6}$ as learning rate. For hyperparameter search, we pick the hyperparameters which work best on a development set that has been generated in the same way as the training set.

We implemented conceptual similarity in Python using standard packages such as PyTorch\textsuperscript{4} for neural networks, HuggingFace transformers library\textsuperscript{5} for BERT and GPT2.

A.2 Parameter Configurations of Expanders

To generate the dataset used in the experiments of this paper, we use all the expansion techniques described in Section 3.2. For each technique, we use different parameter configurations to increase the diversity of the generated expansions. For example, GloVe and Paragram embeddings do not generate the same words given that each embedding model has been trained differently, and thus encode the representation of words in a unique way. Also, in Language Generation Expansion, we use different language models with different allowed lengths. This is to enable the generation of $n$-grams, in addition to words. We give the list of the expanders we use, and their parameters in Table 4.

We have a total of 28 different expanders. We set the parameter $\text{min\_consensus\_rate}$ to 0.3. Consequently, for a new token to be included in the final set of expansions and passed down to the subsequent steps of the dataset generation pipeline (see Section 3 and Figure 1), the token has to be suggested by at least 30% of expanders (9 different expanders in this case). We selected this value by doing a manual hyperparameter search over the following values of $\text{min\_consensus\_rate}$: \{0.1, 0.2, 0.3, 0.4, 0.5, 0.6, 0.7, 0.8, 0.9, 1.0\}. We took the value (i.e. 0.3) that maximized the quality of the final generated dataset, as evaluated in Section 5.2. However, we chose the parameters of the respective expansion techniques manually without conducting a hyperparameter search for the following reasons: (1) There are too many parameters to test, which would make the search space exponentially larger, and thus expensive to explore. (2) The parameter selection of expansion techniques is subjective by nature. We manually chose the parameters such that they make sense (e.g. a negative capacity in ConceptNet Expansion or a very large $top_k$ in Masked Language Modeling Expansion would not be useful), and such that the final expanders would generate a diverse set of expansions from a limited lexicon of seeds.

A.3 Concepts Used in this Work and their Seeds

We select 9 different concepts to include in the conceptual similarity model described in the experiments. We base our choice of concepts on substantial research in behavioral psychology (Moura et al., 2017) whose authors surveyed restaurant seekers and asked them about which factors influence their

\textsuperscript{4}https://github.com/pytorch/pytorch

\textsuperscript{5}https://github.com/huggingface/transformers
In Table 4, we describe the concepts that we use, and give their corresponding seeds for aspects and opinions.

<table>
<thead>
<tr>
<th>WordNet Expansion</th>
<th>num_synsets</th>
<th>hyponym</th>
<th>meronym</th>
<th>hypernym</th>
<th>sisters</th>
</tr>
</thead>
<tbody>
<tr>
<td>3</td>
<td>true</td>
<td>true</td>
<td>true</td>
<td>true</td>
<td>true</td>
</tr>
<tr>
<td>10</td>
<td>true</td>
<td>true</td>
<td>true</td>
<td>false</td>
<td></td>
</tr>
<tr>
<td>5</td>
<td>true</td>
<td>false</td>
<td>true</td>
<td>true</td>
<td></td>
</tr>
</tbody>
</table>

<table>
<thead>
<tr>
<th>ConceptNet Expansion</th>
<th>capacity</th>
<th>minimum_weight</th>
<th>second_level_expansion</th>
</tr>
</thead>
<tbody>
<tr>
<td>3</td>
<td>2.0</td>
<td>true</td>
<td></td>
</tr>
<tr>
<td>5</td>
<td>3.0</td>
<td>true</td>
<td></td>
</tr>
<tr>
<td>10</td>
<td>1.0</td>
<td>false</td>
<td></td>
</tr>
</tbody>
</table>

<table>
<thead>
<tr>
<th>Word Embedding Expansion</th>
<th>embedding_model</th>
<th>num_words</th>
<th>distance_metric</th>
</tr>
</thead>
<tbody>
<tr>
<td>Word2vec</td>
<td>20</td>
<td>euclidean distance</td>
<td></td>
</tr>
<tr>
<td>GloVe</td>
<td>20</td>
<td>cosine similarity</td>
<td></td>
</tr>
<tr>
<td>Fasttext</td>
<td>20</td>
<td>euclidean distance</td>
<td></td>
</tr>
<tr>
<td>Paragram</td>
<td>20</td>
<td>cosine similarity</td>
<td></td>
</tr>
<tr>
<td>ConceptNet</td>
<td>20</td>
<td>euclidean distance</td>
<td></td>
</tr>
</tbody>
</table>

<table>
<thead>
<tr>
<th>Language Generation Expansion</th>
<th>model</th>
<th>top_k</th>
<th>max_length</th>
<th>num_beams</th>
</tr>
</thead>
<tbody>
<tr>
<td>GPT2</td>
<td>20</td>
<td>1</td>
<td>200</td>
<td></td>
</tr>
<tr>
<td>GPT2</td>
<td>20</td>
<td>2</td>
<td>200</td>
<td></td>
</tr>
<tr>
<td>T5 base</td>
<td>20</td>
<td>3</td>
<td>200</td>
<td></td>
</tr>
<tr>
<td>T5 base</td>
<td>10</td>
<td>3</td>
<td>50</td>
<td></td>
</tr>
</tbody>
</table>

<table>
<thead>
<tr>
<th>Masked Language Modeling Expansion</th>
<th>model</th>
<th>top_k</th>
</tr>
</thead>
<tbody>
<tr>
<td>BERT base</td>
<td>10</td>
<td></td>
</tr>
<tr>
<td>BERT base</td>
<td>20</td>
<td></td>
</tr>
<tr>
<td>BERT large</td>
<td>10</td>
<td></td>
</tr>
<tr>
<td>BERT large</td>
<td>20</td>
<td></td>
</tr>
<tr>
<td>RoBERTa large</td>
<td>10</td>
<td></td>
</tr>
<tr>
<td>RoBERTa large</td>
<td>20</td>
<td></td>
</tr>
<tr>
<td>ALBERT large</td>
<td>10</td>
<td></td>
</tr>
<tr>
<td>ALBERT large</td>
<td>20</td>
<td></td>
</tr>
</tbody>
</table>

Table 4: The full list of expansion techniques and their parameter configurations that we used to expand the seed words in our experiments.
<table>
<thead>
<tr>
<th>Concept</th>
<th>Aspects</th>
<th>Opinions 1 (Good)</th>
<th>Opinions 2 (Bad)</th>
<th>Opinions 3 (Healthy)</th>
<th>Opinions 2 (Creative)</th>
</tr>
</thead>
<tbody>
<tr>
<td><strong>Price</strong></td>
<td>price, cost, payment</td>
<td>low, good, fair, acceptable, cheap, not too expensive, affordable, great</td>
<td>expensive, exaggerated, costly, overpriced, high, pricy</td>
<td></td>
<td></td>
</tr>
<tr>
<td><strong>Food</strong></td>
<td>food, menu, plate, cuisine, meal, lunch, dinner, breakfast, cooking, snack, beverage, drink, pizza, pasta, chicken, meat, steak, rice, soup, dessert, dish, fish, salad</td>
<td>tasty, good, excellent, succulent, okay, delicious, well seasoned, perfectly cooked</td>
<td>bad, flavorless, bland, not seasoned, cold, disgusting, unappetizing, flat, gross, boring, awful, terrible, dry</td>
<td>healthy, organic, high quality, fresh</td>
<td>novel, interesting, creative</td>
</tr>
<tr>
<td><strong>Service</strong></td>
<td>staff, waiter, waitress, cashier, service</td>
<td>friendly, smiling, good, helpful, likable</td>
<td>knowledgable, quick, fast, efficient, high quality, professional</td>
<td>grumpy, horrible, slow, irritating, bad</td>
<td></td>
</tr>
<tr>
<td><strong>Cleaning</strong></td>
<td>place, hygiene, kitchen, bathroom, utensils, plates, cutlery, silverware, trays, dishes, table, chair, furniture</td>
<td>clean, impeccable, bright, lavish, luxurious, washed, shining</td>
<td>dirty, bad, in bad shape, stained, greasy, not washed, poor, disgusting</td>
<td></td>
<td></td>
</tr>
<tr>
<td><strong>Parking</strong></td>
<td>parking, parking lot, parking area, parking convenience, parking space</td>
<td>free, available, empty, safe, large</td>
<td>unavailable, poor, narrow, small, hard to find</td>
<td></td>
<td></td>
</tr>
<tr>
<td><strong>Environment</strong></td>
<td>place, environment, setting, surroundings, decor, lighting, music, ventilation, furniture, air conditioning, air conditioner</td>
<td>good, excellent, great, cozy, comfortable, sophisticated, good taste, pleasant, memorable, adequate, beautiful, soothing, calming, fancy, attractive, happy, relaxing, nice, charming</td>
<td>bad, horrible, bad taste, uncomfortable, dark, noisy, terrible, crowded, sad, depressing, boring</td>
<td></td>
<td></td>
</tr>
<tr>
<td><strong>Location</strong></td>
<td>location, area, place, address</td>
<td>near, good, downtown, lively, touristy, popular, secure, safe, good, trustable</td>
<td>far, bad, polluted, remote, dark, unsafe, unsecure, dangerous</td>
<td></td>
<td></td>
</tr>
<tr>
<td><strong>Ambiance</strong></td>
<td>ambiance, atmosphere, air, experience, environment, setting, decor, lighting, music, ventilation, furniture</td>
<td>cozy, good, excellent, romantic, nice, upscale, trendy, loved, enjoyed, fun</td>
<td>horrible, terrible, disgusting, bad, not good, disappointing, noisy, dark, depressing, boring</td>
<td></td>
<td></td>
</tr>
<tr>
<td><strong>Menu</strong></td>
<td>menu, selection, list, choice, choices, option, options</td>
<td>wide, large, varied, variety, good, excellent, creative</td>
<td>small, shabby, narrow, bad</td>
<td></td>
<td></td>
</tr>
</tbody>
</table>