# McPhraSy: Multi-Context Phrase Similarity and Clustering

**Amir DN Cohen**[1,2] [*], **Hila Gonen**[3] [†], **Ori Shapira**[2],
**Ran Levy**[2], **and Yoav Goldberg**[1,4]

[1]Bar-Ilan University [2]Amazon [3]University of Washington
[4]Allen Institute for Artificial Intelligence

{amirdnc, hilagnn, yoav.goldberg}@gmail.com
{orishap, ranlevy}@amazon.com

## Abstract

Phrase similarity is a key component of many NLP applications. Current phrase similarity methods focus on embedding the phrase itself and use the phrase context only during training of the pretrained model. To better leverage the information in the context, we propose McPhraSy (**M**ulti-**c**ontext **Phra**se **S**imilarit**y**), a novel algorithm for estimating the similarity of phrases based on multiple contexts. At inference time, McPhraSy represents each phrase by considering multiple contexts in which it appears and computes the similarity of two phrases by aggregating the pairwise similarities between the contexts of the phrases. Incorporating context during inference enables McPhraSy to outperform current state-of-the-art models on two phrase similarity datasets by up to 13.3%. Finally, we also present a new downstream task that relies on phrase similarity – keyphrase clustering – and create a new benchmark for it in the product reviews domain. We show that McPhraSy surpasses all other baselines for this task.

## 1 Introduction

Estimating similarity between phrases is an important intermediate component for many NLP tasks like question answering (Seo et al., 2018) and machine translation (Ramisch et al., 2013).

As opposed to previous work (e.g., Pennington et al., 2014; Li et al., 2022; Wang et al., 2021) that use the phrase context only during training, we propose McPhraSy – **M**ulti-**c**ontext **Ph**rase **S**imilarit**y** – a novel method for estimating phrase similarity that leverages multiple contexts during inference to improve accuracy. McPhraSy produces a set of representations for a phrase, based on sentences in which it appears. To measure similarity of two phrases, they are compared according to their sets of representations with an innovative technique. Indeed our approach achieves the new state-of-the-art results on two phrase-similarity benchmarks: Turney (Turney, 2012) and PPDB-filtered (Wang et al., 2021) datasets.

In this work we focus on relatively short phrases (noun phrases of 2-3 words) that represent common use cases for output of many NLP applications (e.g. question answering, named entity recognition and relation extraction). Furthermore, we choose to use a relatively strict interpretation of *similarity* between phrases, and we aim to assign high similarity scores to phrases that describe **very** similar concepts, and not just related ones. For example, "charging cable" and "electronic device" are considered related but not similar. Even phrases with lexical overlap such as "craft project" and "craft room" would not be considered similar. On the other hand, phrase pairs such as "quick delivery" and "super fast shipping" are considered similar.

To explore the aforementioned setting in more depth, we introduce a practical use-case of phrase similarity – keyphrase clustering in the domain of product reviews. This task is useful both as an intermediate step for other tasks (e.g. aspect based summarization) and as a downstream task (e.g. given a product keyphrase, retrieve the reviews that mention phrases similar to it).

We curate a dataset for this newly introduced keyphrase clustering task with careful collection of phrases and manual annotation of clusters. We then demonstrate that McPhraSy achieves impressive results on this task compared to all baselines.

The main contributions of this work can be summarized as follows: (1) We demonstrate that existing phrase similarity methods lack information found in phrase contexts at inference time. We overcome this shortcoming by proposing a novel method for phrase similarity estimation. Our method not only leverages the context at inference time, but also takes into account multiple contexts

[*] Completed as part of an internship at Amazon.
[†] Conducted during a post-doc at Amazon.



*Findings of the Association for Computational Linguistics: EMNLP 2022*, pages 3538 - 3550
December 7-11, 2022 

per phrase to make the results more robust; (2) We propose a new downstream task – keyphrase clustering – which relies on phrase-based similarity, and curate an evaluation dataset in the domain of product reviews; (3) We demonstrate that our proposed method outperforms all other baselines on both phrase similarity benchmarks and the keyphrase clustering task.[1]

## 2 Related Work

**Representation of text spans.** Producing word-level representations has been extensively explored (Brown et al., 1992; Bengio et al., 2000; Turian et al., 2010; Collobert et al., 2011; Mikolov et al., 2013b; Pennington et al., 2014; Peters et al., 2018; Devlin et al., 2019), mainly using the surroundings of the word as a signal for its meaning.

Representing text spans larger than one word can be achieved by non-trivial combinations of word representations (e.g., Yu and Dredze, 2015; Wieting et al., 2016; Arora et al., 2017; Chang et al., 2021). However, these approaches use the phrase context only during the pretraining stage and are consequently less effective for capturing the compositional meaning of the span (Yu and Ettinger, 2020).

Phrase-level embeddings assist in tasks such as paraphrase detection, question-answering (QA), and topic modeling, and are often generated in compliance with the task objective. Wang et al. (2021) fine-tune BERT (Devlin et al., 2019) with synthetically generated paraphrases to detect lexically-differing text similarities. Lee et al. (2021) optimize SpanBERT (Joshi et al., 2020) for QA to link a question to a phrasal answer within a passage. Li et al. (2022) and Zhou and Wakabayashi (2022) apply contrastive learning over LUKE (Yamada et al., 2020) or a combination of Sentence-BERT (Reimers and Gurevych, 2019) and Phrase-BERT (Wang et al., 2021), for the use of clustering together topically related phrases within a corpus. These methods for rendering phrase representations leverage a single sentence-level context of the phrase, only while training. In contrast, our approach takes advantage of *many* contexts at *inference time* in order to capture the meaning of the phrase.

**Phrase clustering.** Grouping related phrases together is useful for various tasks, and previous work has thus assessed phrase clustering on task-specific data. SanJuan and Ibekwe-SanJuan (2006) cluster tens of thousands of out-of-context scientific phrases to a predetermined number of categories. Kuhn et al. (2010) cluster source and target language n-grams to assist in the sentence translation task, evaluating the clustering only extrinsically through translation quality. Lin and Wu (2009) apply phrase clustering for named entity recognition and query classification and evaluate correspondingly on relevant datasets. Zhou and Wakabayashi (2022) manually evaluate grouped phrases to assess topic-relatedness, and Li et al. (2022) carry out coarse-grained phrase classification on respective datasets. To the best of our knowledge, we introduce the first phrase clustering dataset (§4), which contains a variable number of clusters in each phrase grouping. This novel dataset allows explicit evaluation with clustering metrics.

## 3 The `McPhraSy` Method

While most existing phrase embedding methods, such as averaged GloVe (Pennington et al., 2014) word embeddings and Phrase-BERT (Wang et al., 2021), rely solely on the phrase to create a representation in inference time, our approach relies on multiple contexts in which the given phrase appears. For example, given the phrase "*brunette hair*", previous methods only consider the standalone phrase, while `McPhraSy` places it in multiple contexts (masked sentences) such as, "*This is a great shampoo for* [MASK]." or "*I always wanted to have* [MASK]*, and thanks to this hair dye I'm now a brunette!*". The context of a word (or phrase) is known to be highly indicative of its meaning (Firth, 1957). We hypothesize that current models do not succeed in learning the full meaning from the context of phrases during training. By using contexts directly at *inference* time, we hope to improve phrase representations.

Given two phrases, `McPhraSy` retrieves contexts for both phrases, extracts the vector representation of each context (§3.1) assisted by a trained model (§3.2), and applies a similarity function between the two sets of representations (§3.3).

### 3.1 Representing a Phrase

Given a phrase $p$, we start by searching for $p$ in an unlabeled corpus of raw text and retrieve $m$ sentences in which it appears. Each sentence is masked at the position of $p$ using a single [MASK]

[1] The dataset and code will be available online.

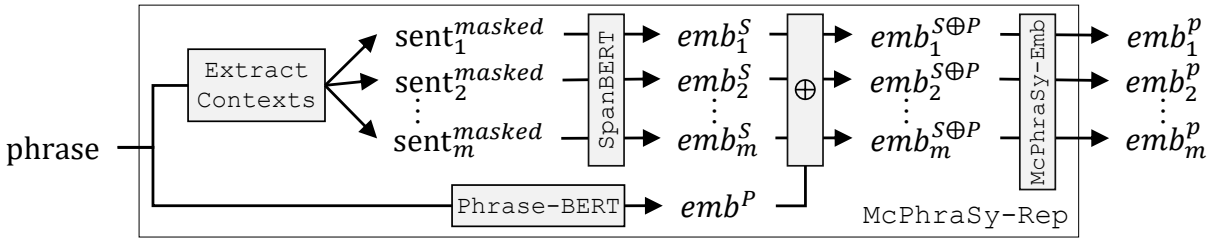



Figure 1: The process for producing the phrase representations with McPhraSy. SpanBERT and Phrase-BERT vectors are concatenated and fed into the McPhraSy embedder to create the final representations.

token and is passed to SpanBERT to get the representation at the mask position. The set of (up to $m$) corresponding outputs is used as the initial representation of $p$. Each such vector (of dimension $d_{sbert} = 768$) is concatenated with the Phrase-BERT embeddings of the phrase (of dimension $d_{pbert} = 768$), and is then given as input to the McPhraSy embedder, a multi-layer perceptron (MLP) trained to output the final phrase representation (of dimension $d_{\texttt{McPhraSy}} = 768$). This process, depicted in Figure 1, results in a set of representations for phrase $p$.

### 3.2 Training a Dedicated Model for Phrase Representation

The McPhraSy embedder is a 2-layer MLP that receives an initial representation of a phrase, and produces a finetuned representation for the phrase. As illustrated in Figure 2, this model is trained using triplet-loss, a commonly used loss function for learning representations based on similarities. In our case, it requires representing three phrases: (a) an anchor phrase $p_a$, using a random context $c_{p_a}$; (b) a positive example $p_+$, which is the same phrase $p_a$ but in a different context $c_{p_+}$; and (c) a negative example $p_-$ of a different (random) phrase, with context $c_{p_-}$. For example, if $p_a$ = "hair color", then $c_{p_a}$ = "this hair color is very bright", $c_{p_+}$ = "this hair color ruined my hair!", and $c_{p_-}$ = "this battery lasts for three hours".

The loss function is defined as:

$$\texttt{triplet-loss}(p_a, p_+, p_-) = \max(\mathrm{d}(e(c_{p_a}), e(c_{p_+})) - \mathrm{d}(e(c_{p_a}), e(c_{p_-})) + \alpha, 0) \quad (1)$$

where $\mathrm{d}(\cdot,\cdot)$ is a distance metric (e.g., $l_2$ or cosine distance), $\alpha$ is a margin hyper-parameter, that encourages preference of positive over negative instances, and $e(\cdot)$ is the embedding function. The minimization of Equation 1 requires the embedding function $e$ to assign close embeddings for $p_a$ and $p_+$, and more distant ones for $p_a$ and $p_-$.

To generate training triplets $(p_a, p_+, p_-)$, we use *hard sampling* (Hermans et al., 2017), which aims to find the contexts for $p_+$ and $p_-$ that are the closest to a context of $p_a$, in order to challenge the learned function with more difficult classifications (see Appendix A for additional technical details).

The underlying models that generate the initial representations (in our case, SpanBERT and Phrase-BERT models) are kept frozen during training.

While we show that using McPhraSy with pretrained SpanBERT model as the sole initial representation can already surpass current state-of-the-art on some similarity tasks (§5.2.2), we empirically find that integrating the phrase representation (like with Phrase-BERT) is beneficial for keyphrase clustering (§5.3.2). McPhraSy harnesses the advantages of both the contextual meaning of the phrase from SpanBERT, as well as the generic meaning of the phrase on its own from Phrase-BERT. Incorporating Phrase-BERT is not a trivial operation since phrase $p_a$ is the same as $p_+$ but different from $p_-$. A simple concatenation of Phrase-BERT embeddings to the context representation would make it trivial for the model to differentiate between the positive and negative examples, making the training redundant. Instead, when training the McPhraSy encoder, we mask the Phrase-BERT embedding (with zeros) for $p_a$, essentially integrating Phrase-BERT only for $p_+$ and $p_-$ (see Figure 2).

### 3.3 Estimating Similarity

We next wish to use the aforementioned multi-context-based representations in order to determine the similarity between two phrases $p_a$ and $p_b$.

A naive approach would be to average the multiple-context representations of $p_a$ and $p_b$ respectively, and compute the cosine similarity between the two averaged vectors. However, we propose an enhanced method for phrase similar-

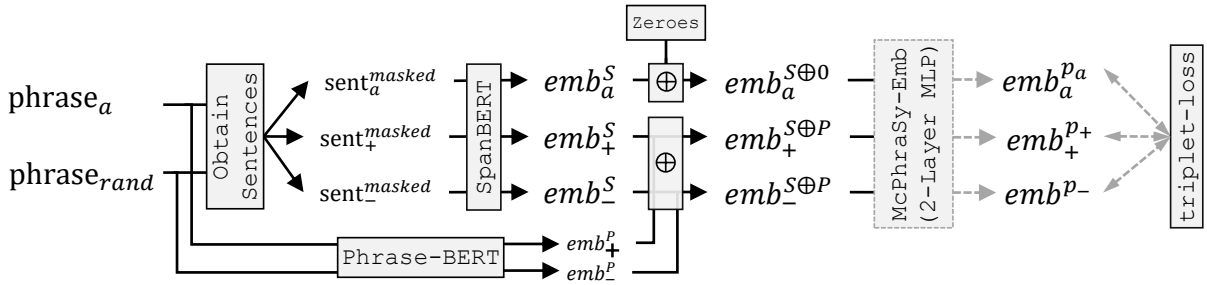



Figure 2: The process for training the McPhraSy embedder. Plus (+) and anchor ($a$) correspond to the samples that share a common masked phrase, while minus ($-$) corresponds to a sample with a different masked phrase. In this process, the McPhraSy-embedder is trained using the triplet loss.

ity based on context analysis of the two phrases, which achieves improved accuracy over the aforementioned averaging approach, as shown in Section 5.2. This method is based on the intuition that the more contexts are similar across the two phrases, the more the phrases themselves are similar in meaning.

Let $c_p \in C_p$ be a context of phrase $p$ and $e(c_p)$ the representation of $c_p$ (extracted as described in Section 3.1), and let

$$d^{p'}_{c_p} = \min_{c_{p'} \in C_{p'}} dist(e(c_p), e(c_{p'}))$$

$$d^{p}_{c_p} = \min_{\bar{c_p} \in C_p \setminus \{c_p\}} dist(e(c_p), e(\bar{c_p}))$$

i.e., the distance from $c_p$ to the closest context of phrase $p' \neq p$, and to a different context of $p$, respectively. Function $dist(\cdot,\cdot)$ measures distance between vectors (we use cosine distance throughout the paper). Then, given two phrases $p$ and $p'$, we define the difference function $\delta(c_p, p')$ as:

$$\delta(c_p, p') = d^{p'}_{c_p} - d^{p}_{c_p}$$

i.e., the difference between the distances from $c_p$ to the two close contexts, of $p$ and $p'$.

Now, let $D_{p,p'}$ be the random variable that represents all differences $\delta(c_p, p')$ and $\delta(c_{p'}, p)$ for random context $c_p \sim C_p$ and $c_{p'} \sim C_{p'}$.

The more $p$ and $p'$ are similar, the more concentration of probability will amass near 0, since the close context representations between the phrases will yield small differences. To assess this distance in a quantitative way, we define the cumulative distribution function (CDF) of $D_{p,p'}$ to be $f_{p,p'}$, where

$$f_{p,p'}(x) = \mathrm{P}\left(D_{p,p'} < x\right)$$

is the probability that a difference between close contexts is less than $x$ (for $dist(\cdot,\cdot)$ being cosine distance, $-2 \leq x \leq 2$).

Finally, using $f_{p,p'}$ we define the similarity of $p$ and $p'$ to be:

$$\texttt{McPhraSy-sim}(p, p') = \int_{-2}^{2} f_{p,p'}(x)\, dx$$

The intuition behind this similarity is that similar phrases likely have close context representations, while dissimilar phrases have a higher chance of having distant representations.

To estimate $f_{p,p'}$, we sample $D_{p,p'}$ by activating $\delta$ on each of the available contexts. Based on these samples, we calculate the empirical distribution of $D_{p,p'}$ by generating the histogram of differences, and the empirical CDF of $D_{p,p'}$ by summation over this histogram. Empirically we find that $x$ predominantly falls between 0 and 1, and therefore compute the integral in that range.[2]

Note that the step for extracting and preparing the phrase representations can be relatively computationally expensive, however, the similarity estimation step is quick. Therefore, to use McPhraSy, one can prepare the representations offline in a one-time preprocessing step,[3] and then evaluate similarities efficiently upon request.

## 4 Phrase-Based Clustering

In addition to standard phrase similarity benchmarks, the ability to compare the likeness of phrases can be extrinsically evaluated by means of phrase *clustering*. Joining and separating phrases within a set adds a level of complexity that requires a more fine-grained ability to measure phrase similarity, as a bad estimation for a single phrase pair may result in very different allocations to clusters.

[2] Empirically, McPhraSy-sim values are in $[0, 0.25]$.

[3] In addition, contexts for each phrase can be made easily retrievable by using inverted indexes over the corpus.

| Dataset Stats | w/ Sing. | w/o Sing. |
|---|---|---|
| # groups | 106 | 106 |
| phrases per group | 25 | 25 |
| avg. # clusters per group | 16.95 | 3.52 |
| avg. cluster size | 1.47 | 3.17 |
| avg. # overlapping clusters | 2.54 | 0.90 |

Table 1: Statistics on our clustering dataset, shown when including or excluding singleton (Sing.) clusters. The last row shows the number of clusters that contain words from a separate cluster within the same group, portraying the lexical overlap between clusters in a group.

Moreover, the phrase clustering task serves practical applications such as identification of reoccurring elements within documents or product reviews for use of information extraction or summarization. While there exist datasets that cluster noun phrases to predetermined cluster categories, like types of named entities (e.g., Tjong Kim Sang and De Meulder, 2003; Derczynski et al., 2017), to the best of our knowledge, we are the first to establish a phrase clustering dataset without fixed classes.

### 4.1 Curating a Dataset

Our phrase clustering dataset is based on the Amazon Review Dataset released by Ni et al. (2019). It consists of 106 groups of 25 noun phrases each, spanning 5 different categories,[4] and each group comprises clusters of similar phrases. Full statistics are available in Table 1. A cluster either contains phrases equivalent in meaning (e.g., "assorted colors" and "variety of colors") or phrases that describe the same of a kind (e.g., "business cards", "place cards", and "time cards"). Precedence is given for clustering equivalent phrases before phrases that are the same of a kind.

To create the dataset we first collect groups of noun phrases and then annotate clusters within each group.

**Collecting groups of phrases.** A group constitutes a *seed phrase* and 24 related phrases and is created as follows.

First, for each *product* in a category, we extract the top 30 most frequent noun phrases of 2-3 words from all the available reviews of the product. Then, to extract the top 2000 most frequent phrases of a *category*, we aggregate the phrase counts from products in that category. Some undesired phrases are then filtered out with basic lexical heuristics (see Appendix B.1 for details).

Next, for each category, we select the most frequent phrase as the seed phrase. We form a group around the seed phrase by selecting, from the frequent phrases in the category, the 24 phrases that are most similar to it according to cosine similarity of their word2vec (Mikolov et al., 2013a) representations (average of the words in the phrase).[5] Applying word2vec similarity yields many lexically similar phrases within a group that are not necessarily similar in meaning, thus providing a challenging case for clustering. Then, we continue iteratively to the next most frequent seed phrase in the category and keep the new group only if its intersection with each of the previous groups does not include more than half of the phrases. We prepared 106 such groups of 25 phrases over all categories.

**Forming clusters within groups.** To cluster each of the phrase groups, we first experimented with various crowdsourcing tasks, which yielded overly noisy results (see Appendix B.2 for more details). We therefore turned to internal manual annotation, in which three authors of this paper participated. First, six groups were annotated by all three annotators, resulting in average clustering agreement scores of 0.93 V-measure, 0.57 Adjusted Rand, and 0.62 Adjusted Mutual Information (see §5.3.1 for metric explanations), after which differences were reconciled. The high agreement and reconciliation process permitted a single annotation for the rest of the groups. To that end, the 100 remaining groups were divided amongst the three annotators. Given a group of phrases, the annotator first clustered together phrases that are equivalent in meaning, and then from the remaining phrases clustered those that are the same of a kind.

Additional technical details on the dataset creation process are available in Appendix B.

### 4.2 Qualitative Examination

In addition to the distinction of cluster types (*equivalent meanings* versus *same of a kind*), many phrases within a group lexically overlap, but may or may not be clustered together, thus further challenging a clustering algorithm to distinguish between such cases. Moreover, each group contains many singleton clusters, i.e., when a cluster consists of one phrase only, again contributing to the difficulty

[4] We use categories "Arts Crafts and Sewing", "Automotive", "Grocery and Gourmet Food", "Office Products" and "Patio Lawn and Garden".

[5] Using spaCy (Honnibal et al., 2020).

of deciding on what phrases to cluster together. As apparent in the last row of Table 1, a cluster shares (non stop-) words with an average of 2.5 other clusters, including singleton clusters.

Indeed, we find plenty of examples that demonstrate the challenges posed. For instance, the phrases "hot chocolate", "hot tea" and "hot coffee" are clustered together, but not with "hot cereal", "hot sauce", or even "hot water". There are also cases of non-lexically overlapping phrases in a cluster, such as "quilt shop" and "fabric store", with a lexical overlap in a different cluster, such as one containing "jewelry store".

We also find situations where context is essential for deciding whether phrases should be clustered. For example, "backing plate", "pressure plate" and "skid plate" might seem similar-kinded, however, they are unrelated kinds of plates with different functions. Automotive context is required to apprehend the meanings.

Another phenomenon is when phrases would be clustered differently in different groups due to the precedence of cluster types. For example, "fit right", "fit well", "fit properly", "decent fit" and "fit as described" will cluster separately from "does not fit", "terrible fit" and "poor fit", since phrases in each separate cluster are equivalent in meaning. However if only "decent fit" and "poor fit" were in a group, they might cluster together because they are the same of a kind.

## 5 Experiments

We first evaluate McPhraSy on phrase similarity benchmarks, and then continue to examine its utility for clustering on our new phrase-based clustering dataset.

### 5.1 Training McPhraSy

Our training data consists of 27,723 phrases with their contexts, sampled from Wikipedia using the Spike platform (Shlain et al., 2020). Overall we sample 24,361,780 sentences for training the 2-layer MLP and use a maximum of 300 contexts per phrase (more details in Appendix A.2). Freezing the underlying SpanBERT and Phrase-BERT models has two benefits: (a) it allows using a relatively large batch size of 300 sentences; (b) freezing lower layers of the model has a regularization effect on the overall model.

### 5.2 Phrase Similarity

#### 5.2.1 Experimental Setup

**Datasets.** We adopt the **PPDB-filtered** dataset developed by Wang et al. (2021) that was devised for pairwise phrase similarity assessment. This data is based on the PPDB 2.0 dataset (Pavlick et al., 2015), with filtration heuristics proposed by Yu and Ettinger (2020). The PPDB-filtered dataset contains 19,416 phrase pairs, marked as similar or non-similar, with an average phrase-length of about 2.5 words. The phrase pairs are controlled for lexical similarity by assuring that positive (similar) and negative (non-similar) pairs have an identical amount of word overlap. Moreover, phrase pairs are controlled for word biases so that certain words do not overlap considerably more in a particular class. Figure 3 shows an example of similar and dissimilar phrases from PPDB-filtered, with a visualization of the dimension-reduced McPhraSy representations (using PCA; Abdi and Williams, 2010). We used the same dataset split as Wang et al. (2021).

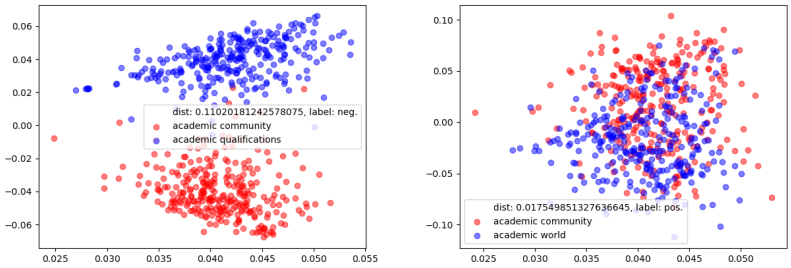
The **Turney** dataset (Turney, 2012) consists of 2,180 groups, where each group is built of a bigram and five unigrams. The accompanying task is to select the unigram that is closest in meaning to the respective bigram. For example, given the bigram "bass fiddle", and the candidates: "contrabass", "pitch", "violin", "speedway", "snood" the model should return "contrabass". Both PPDB and Turney datasets use accuracy as the evaluation metric.

**Baselines.** We evaluate McPhraSy against other methods of phrase or text similarity. Most methods produce representations for the phrases and use cosine similarity to estimate the similarity between them. For phrase representation, we use **GloVe** (Pennington et al., 2014) and **BERT** (Devlin et al., 2019) embeddings of the words in a phrase, as well as **SpanBERT**[6] (Joshi et al., 2020), **Phrase-BERT** (Wang et al., 2021), **Sentence-BERT** (Reimers and Gurevych, 2019) and joint-Sentence-BERT-based (Zhou and Wakabayashi, 2022) (denoted **joint-SB**) representations.

#### 5.2.2 Results

We examine four different variations of McPhraSy. **McPhraSy** is the full model. **McPhraSy SpanBERT+emb** uses the complete model but does

[6] SpanBERT used similar in fashion to (Wang et al., 2021).




(a) Dissimilar phrases

(b) Similar phrases

Figure 3: A 2-dimensional version, reduced with PCA, of the McPhraSy representations of similar phrases (3b) and dissimilar phrases (3a) from the PPDB-filtered dataset.

| Method | Turney | PPDB-filt |
|---|---|---|
| GloVe | 37.8 | 44.2 |
| BERT | 42.6 | 60.1 |
| SpanBERT | 38.7 | 57.3 |
| Sentence-BERT | 51.8 | 64.2 |
| Phrase-BERT | 57.2 | 68.0 |
| joint-SB | 58.0 | - |
| McPhraSy+emb average | 70.7* | 68.5 |
| McPhraSy SpanBERT only | 70.3* | 68.5 |
| McPhraSy SpanBERT+emb | **71.3*** | **68.7*** |
| McPhraSy | 67.7* | 68.6* |

Table 2: The accuracy (in %) of each method on phrase similarity datasets. PPDB-filt requires a binary choice, and Turney requires a choice from five options. McPhraSy outperforms all baselines. A * signifies significant improvement over the previous state of the art, with $p < 0.05$.

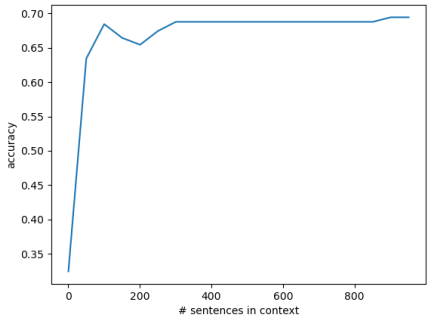



Figure 4: The accuracy on PPDB-filtered as a function of the number of contexts used by McPhraSy to represent phrases (at inference time).

not use Phrase-BERT information during inference (Phrase-BERT embeddings are replaced with zeros). **McPhraSy+emb average** uses the McPhraSy SpanBERT+emb model, but with averaging of vectors instead of using the McPhraSy similarity function (§3.3), i.e., representations of a phrase are averaged and compared to another phrase's average with cosine similarity. **McPhraSy SpanBERT only** uses the McPhraSy similarity on SpanBERT representations only (without the trained embedding model, nor the Phrase-BERT representations).

Table 2 presents the results of the different phrase similarity methods on the two benchmarks. We report the overall percent of accurate predictions, where Turney has 5 alternative choices per instance, and PPDB-filtered requires a binary prediction. McPhraSy significantly improves over the baselines on both datasets (with $p < 0.05$). While all four versions of McPhraSy surpass current state-of-the-art models by a large margin on Turney, the most substantial improvement of 13.3 points is achieved by McPhraSy SpanBERT+emb.

We find that the McPhraSy SpanBERT-only version surpasses the current state-of-the-art model (joint-SB) by a large margin (12.7 points). Adding the extra embedding model contributes substantially, but surprisingly, concatenating the Phrase-BERT embeddings reduces performance compared to the McPhraSy SpanBERT+emb model. We attribute this behavior to the nature of the Turney dataset which emphasizes semantic similarity without much lexical overlap. McPhraSy SpanBERT only doesn't use any lexical information, and adding lexical information (using Phrase-BERT) degrades the model accuracy on Turney.

**Number of contexts.** We assess the effect of the number of contexts ($m$) on the overall accuracy of McPhraSy on the PPDB-filtered dataset. As Figure 4 indicates, most of the improvement is accredited to the first 100 contexts, with a slight increase until 300. Adding contexts beyond 300 seems to have a marginal effect.

### 5.3 Phrase Clustering

We now turn to evaluate some of the phrase similarity methods as the underlying functions for the clustering task.

#### 5.3.1 Experimental Setup

**Baselines.** We employ the Agglomerative Clustering method with complete linkage,[7] and provide it with the pairwise phrase similarity matrix based on the different similarity functions. We compare the use of McPhraSy to **GloVe** or **Phrase-BERT** with cosine similarity, and to character-level **Edit-distance**.[8] In addition, we provide the clustering algorithm with a pre-specified number of clusters to form, denoted $k$. Specifically, we set $k \in \{10, 15, 20, \text{gold}\}$, where gold is the actual number of clusters in the respective group being clustered. The alternative $k$ values were chosen based on the average number of clusters in the groups (~17).

**Clustering metrics.** Standard metrics for measuring clustering quality include V-measure (Rosenberg and Hirschberg, 2007), Adjusted Rand (Hubert and Arabie, 1985) and Adjusted Mutual Information (Nguyen et al., 2010). These metrics enable the comparison of two cluster assignments. Furthermore, since they are symmetric measures, they can be used for measuring agreement of two cluster assignments, as we did during the dataset annotation process (§4.1).

The **V-measure** is the harmonic mean of homogeneity and completeness. Homogeneity is satisfied if each predicted cluster contains only data points that are members of a single gold cluster. Completeness is satisfied if the members of any given gold cluster are elements of the same predicted cluster. The Rand Index considers the amount of data point pairs that are correctly or incorrectly clustered, and **Adjusted Rand** adjusts the value for chance (where a score of 0 reflects the quality of a random solution, and positive or negative scores are better or worse than that). Similarly, **Adjusted Mutual Information** adjusts the Mutual Information score.

#### 5.3.2 Results

Table 3 presents the scores of the clusterings using the different similarity methods, and with different $k$-parameter values. McPhraSy achieves the highest scores across the board, and significantly so in several cases. We find that Edit-distance and GloVe achieve impressive scores, though still lower than Phrase-BERT and McPhraSy.

While McPhraSy variants excel in both similarity and clustering tasks, surprisingly, incorporating Phrase-BERT during inference harmed performance on both similarity datasets, but improved results in the clustering task. We attribute this behavior to the different nature of the phrases in both tasks. In the similarity datasets, special care was taken to reduce lexical similarity between compared phrases. However, in our dataset we do not aim to reduce such similarities. We presume that real world tasks might also require lexical similarity to perform well. This is also highlighted by the poor performance of McPhraSy SpanBERT only, which underperforms (at times even below the naive Edit-distance baseline), likely because it does not have direct access to the phrase.

## 6 Conclusion

In this work we address the task of phrase similarity and propose to add context information to the phrase representation during inference. This is done by extracting representations of different contexts per phrase and aggregating their pairwise similarity with a novel method. We show that McPhraSy surpasses previous SOTA methods for standard phrase similarity benchmarks.

Additionally, we present a new task of phrase-based clustering that relies on high quality phrase similarity estimation. We collect a new dataset for this task in the domain of product reviews and annotate 106 groups with 2650 phrases for cluster assignment. We show that McPhraSy improves over all baselines, thanks to its innovative mechanism that integrates phrase contexts with existing phrase representation models.

Finally, since contexts are considered at inference time, we expect McPhraSy to work smoothly across domains, especially when texts in the new domain are relatively scarce. We leave such experiments for future work.

## 7 Limitations

While our model outperforms strong competing methods, it requires a complementary corpus with a substantial amount of sentences containing the phrases in question. We believe that in real world

[7] Using scikit-learn (Pedregosa et al., 2011).

[8] We apply edit-distance to examine whether simple lexical similarity heuristics can achieve decent clustering of the highly lexically overlapping phrases in a group.

| $k =$ | 10 | | | 15 | | | 20 | | | gold | | |
|---|---|---|---|---|---|---|---|---|---|---|---|---|
| **Sim. Function** | **V** | **AR** | **AMI** | **V** | **AR** | **AMI** | **V** | **AR** | **AMI** | **V** | **AR** | **AMI** |
| Edit distance | 0.78 | 0.27 | 0.33 | 0.84 | 0.27 | 0.32 | 0.87 | 0.16 | 0.20 | 0.87 | 0.28 | 0.32 |
| GloVe | 0.79 | 0.30 | 0.36 | 0.85 | 0.29 | 0.35 | 0.88 | 0.25 | 0.30 | 0.88 | 0.36 | 0.41 |
| Phrase-BERT | **0.81**$^*$ | 0.33 | **0.39**$^*$ | 0.87 | 0.36 | 0.42 | 0.89 | **0.29** | 0.36 | 0.88 | 0.42 | 0.48 |
| McPhraSy SpanBERT only | 0.76 | 0.21 | 0.27 | 0.84 | 0.26 | 0.32 | 0.88 | 0.23 | 0.28 | 0.88 | 0.31 | 0.35 |
| McPhraSy SpanBERT+emb | 0.76 | 0.22 | 0.29 | 0.84 | 0.26 | 0.31 | 0.88 | 0.23 | 0.29 | 0.86 | 0.36 | 0.33 |
| McPhraSy | **0.81**$^*$ | **0.34**$^*$ | **0.39**$^*$ | **0.89**$^*$ | **0.37** | **0.43** | **0.90** | 0.28 | **0.37** | **0.89** | **0.45**$^*$ | **0.50**$^*$ |

Table 3: The V-measure, Adjusted Rand, and Adjusted Mutual Information scores on our noun phrase clusters dataset at different $k$ values (number of clusters), for different similarity function alternatives. When $k =$ gold, $k$ is the actual number of clusters in the respective group. Agglomerative Clustering is employed as the clustering algorithm. Bold values are the highest in their measure, and a $^*$ signifies significant improvement over the next best value, with $p < 0.05$.

applications, where the phrases originate from a corpus anyways, this limitation is somewhat mitigated. In addition, our model requires more compute resources than methods that apply GloVe or Phrase-BERT. However, making such models more efficient is an active research area.

Our new clustering dataset is relatively small scale, consisting of 106 groups of 25 phrases each. The challenge in collecting this data lies in the annotation process. As mentioned, crowdsourcing such a task yielded noisy results that were not suitable for high quality evaluation purposes.

In addition, we use word2vec as part of the data creation (for grouping together phrases). This may inject a certain bias to the dataset in favor of methods that make use of similar-in-nature word embeddings such as GloVe.

## Acknowledgements

This project has received funding from the European Research Council (ERC) under the European Union's Horizon 2020 research and innovation programme, grant agreement No. 802774 (iEXTRACT).

## References

Hervé Abdi and Lynne J. Williams. 2010. Principal Component Analysis. *WIREs Computational Statistics*, 2(4):433–459.

Sanjeev Arora, Yingyu Liang, and Tengyu Ma. 2017. A Simple but Tough-to-Beat Baseline for Sentence Embeddings. In *International Conference on Learning Representations*.

Yoshua Bengio, Réjean Ducharme, and Pascal Vincent. 2000. A Neural Probabilistic Language Model. In *Advances in Neural Information Processing Systems*, volume 13. MIT Press.

Peter F. Brown, Vincent J. Della Pietra, Peter V. deSouza, Jenifer C. Lai, and Robert L. Mercer. 1992. Class-Based n-gram Models of Natural Language. *Computational Linguistics*, 18(4):467–480.

Haw-Shiuan Chang, Amol Agrawal, and Andrew McCallum. 2021. Extending Multi-Sense Word Embedding to Phrases and Sentences for Unsupervised Semantic Applications. *Proceedings of the AAAI Conference on Artificial Intelligence*, 35(8):6956–6965.

Ronan Collobert, Jason Weston, Léon Bottou, Michael Karlen, Koray Kavukcuoglu, and Pavel Kuksa. 2011. Natural Language Processing (Almost) from Scratch. *Journal of Machine Learning Research*, 12:2493–2537.

Leon Derczynski, Eric Nichols, Marieke van Erp, and Nut Limsopatham. 2017. Results of the WNUT2017 Shared Task on Novel and Emerging Entity Recognition. In *Proceedings of the 3rd Workshop on Noisy User-generated Text*, pages 140–147, Copenhagen, Denmark. Association for Computational Linguistics.

Jacob Devlin, Ming-Wei Chang, Kenton Lee, and Kristina Toutanova. 2019. BERT: Pre-training of Deep Bidirectional Transformers for Language Understanding. In *Proceedings of the 2019 Conference of the North American Chapter of the Association for Computational Linguistics: Human Language Technologies, Volume 1 (Long and Short Papers)*, pages 4171–4186, Minneapolis, Minnesota. Association for Computational Linguistics.

John R Firth. 1957. A Synopsis of Linguistic Theory, 1930-1955. *Studies in linguistic analysis*.

Alexander Hermans, Lucas Beyer, and Bastian Leibe. 2017. In Defense of the Triplet Loss for Person Re-Identification.

Matthew Honnibal, Ines Montani, Sofie Van Landeghem, and Adriane Boyd. 2020. spaCy: Industrial-strength Natural Language Processing in Python.

Lawrence Hubert and Phipps Arabie. 1985. Comparing Partitions. *Journal of Classification*, 2(1):193–218.

Mandar Joshi, Danqi Chen, Yinhan Liu, Daniel S. Weld, Luke Zettlemoyer, and Omer Levy. 2020. SpanBERT: Improving Pre-training by Representing and Predicting Spans. *Transactions of the Association for Computational Linguistics*, 8:64–77.

Roland Kuhn, Boxing Chen, George Foster, and Evan Stratford. 2010. Phrase Clustering for Smoothing TM Probabilities - or, How to Extract Paraphrases from Phrase Tables. In *Proceedings of the 23rd International Conference on Computational Linguistics (Coling 2010)*, pages 608–616, Beijing, China. Coling 2010 Organizing Committee.

Jinhyuk Lee, Mujeen Sung, Jaewoo Kang, and Danqi Chen. 2021. Learning Dense Representations of Phrases at Scale. In *Proceedings of the 59th Annual Meeting of the Association for Computational Linguistics and the 11th International Joint Conference on Natural Language Processing (Volume 1: Long Papers)*, pages 6634–6647, Online. Association for Computational Linguistics.

Jiacheng Li, Jingbo Shang, and Julian McAuley. 2022. UCTopic: Unsupervised Contrastive Learning for Phrase Representations and Topic Mining. In *Proceedings of the 60th Annual Meeting of the Association for Computational Linguistics (Volume 1: Long Papers)*, pages 6159–6169, Dublin, Ireland. Association for Computational Linguistics.

Dekang Lin and Xiaoyun Wu. 2009. Phrase Clustering for Discriminative Learning. In *Proceedings of the Joint Conference of the 47th Annual Meeting of the ACL and the 4th International Joint Conference on Natural Language Processing of the AFNLP*, pages 1030–1038, Suntec, Singapore. Association for Computational Linguistics.

Tomás Mikolov, Kai Chen, Greg Corrado, and Jeffrey Dean. 2013a. Efficient Estimation of Word Representations in Vector Space. In *1st International Conference on Learning Representations, ICLR 2013, Scottsdale, Arizona, USA, May 2-4, 2013, Workshop Track Proceedings*.

Tomas Mikolov, Ilya Sutskever, Kai Chen, Greg S Corrado, and Jeff Dean. 2013b. Distributed Representations of Words and Phrases and their Compositionality. In *Advances in Neural Information Processing Systems*, volume 26. Curran Associates, Inc.

Vinh Nguyen, Julien Epps, and James Bailey. 2010. Information Theoretic Measures for Clusterings Comparison: Variants, Properties, Normalization and Correction for Chance. *Journal of Machine Learning Research*, 11:2837 – 2854.

Jianmo Ni, Jiacheng Li, and Julian McAuley. 2019. Justifying Recommendations using Distantly-Labeled Reviews and Fine-Grained Aspects. In *Proceedings of the 2019 Conference on Empirical Methods in Natural Language Processing and the 9th International Joint Conference on Natural Language Processing (EMNLP-IJCNLP)*, pages 188–197, Hong Kong, China. Association for Computational Linguistics.

Ellie Pavlick, Pushpendre Rastogi, Juri Ganitkevitch, Benjamin Van Durme, and Chris Callison-Burch. 2015. PPDB 2.0: Better paraphrase ranking, fine-grained entailment relations, word embeddings, and style classification. In *Proceedings of the 53rd Annual Meeting of the Association for Computational Linguistics and the 7th International Joint Conference on Natural Language Processing (Volume 2: Short Papers)*, pages 425–430, Beijing, China. Association for Computational Linguistics.

F. Pedregosa, G. Varoquaux, A. Gramfort, V. Michel, B. Thirion, O. Grisel, M. Blondel, P. Prettenhofer, R. Weiss, V. Dubourg, J. Vanderplas, A. Passos, D. Cournapeau, M. Brucher, M. Perrot, and E. Duchesnay. 2011. Scikit-learn: Machine Learning in Python. *Journal of Machine Learning Research*, 12:2825–2830.

Jeffrey Pennington, Richard Socher, and Christopher Manning. 2014. GloVe: Global Vectors for Word Representation. In *Proceedings of the 2014 Conference on Empirical Methods in Natural Language Processing (EMNLP)*, pages 1532–1543, Doha, Qatar. Association for Computational Linguistics.

Matthew E. Peters, Mark Neumann, Mohit Iyyer, Matt Gardner, Christopher Clark, Kenton Lee, and Luke Zettlemoyer. 2018. Deep Contextualized Word Representations. In *Proceedings of the 2018 Conference of the North American Chapter of the Association for Computational Linguistics: Human Language Technologies, Volume 1 (Long Papers)*, pages 2227–2237, New Orleans, Louisiana. Association for Computational Linguistics.

Carlos Ramisch, Aline Villavicencio, and Valia Kordoni. 2013. Introduction to the Special Issue on Multiword Expressions: From Theory to Practice and Use. *ACM Trans. Speech Lang. Process.*, 10(2).

Nils Reimers and Iryna Gurevych. 2019. Sentence-BERT: Sentence Embeddings using Siamese BERT-Networks. In *Proceedings of the 2019 Conference on Empirical Methods in Natural Language Processing*. Association for Computational Linguistics.

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)*, pages 410–420, Prague, Czech Republic. Association for Computational Linguistics.

Eric SanJuan and Fidelia Ibekwe-SanJuan. 2006. Phrase Clustering Without Document Context. In *Advances in Information Retrieval*, pages 496–500, Berlin, Heidelberg. Springer Berlin Heidelberg.

Minjoon Seo, Tom Kwiatkowski, Ankur Parikh, Ali Farhadi, and Hannaneh Hajishirzi. 2018. Phrase-Indexed Question Answering: A New Challenge for Scalable Document Comprehension. In *Proceedings of the 2018 Conference on Empirical Methods in Natural Language Processing*, pages 559–564, Brussels, Belgium. Association for Computational Linguistics.

Micah Shlain, Hillel Taub-Tabib, Shoval Sadde, and Yoav Goldberg. 2020. Syntactic Search by Example. In *Proceedings of the 58th Annual Meeting of the Association for Computational Linguistics: System Demonstrations*, pages 17–23, Online. Association for Computational Linguistics.

Erik F. Tjong Kim Sang and Fien De Meulder. 2003. Introduction to the CoNLL-2003 Shared Task: Language-Independent Named Entity Recognition. In *Proceedings of the Seventh Conference on Natural Language Learning at HLT-NAACL 2003*, pages 142–147.

Joseph Turian, Lev-Arie Ratinov, and Yoshua Bengio. 2010. Word Representations: A Simple and General Method for Semi-Supervised Learning. In *Proceedings of the 48th Annual Meeting of the Association for Computational Linguistics*, pages 384–394, Uppsala, Sweden. Association for Computational Linguistics.

Peter D. Turney. 2012. Domain and Function: A Dual-Space Model of Semantic Relations and Compositions. *J. Artif. Int. Res.*, 44(1):533–585.

Shufan Wang, Laure Thompson, and Mohit Iyyer. 2021. Phrase-BERT: Improved Phrase Embeddings from BERT with an Application to Corpus Exploration. In *Proceedings of the 2021 Conference on Empirical Methods in Natural Language Processing*, pages 10837–10851, Online and Punta Cana, Dominican Republic. Association for Computational Linguistics.

John Wieting, Mohit Bansal, Kevin Gimpel, and Karen Livescu. 2016. Towards Universal Paraphrastic Sentence Embeddings. In *4th International Conference on Learning Representations, ICLR 2016, San Juan, Puerto Rico, May 2-4, 2016, Conference Track Proceedings*.

Ikuya Yamada, Akari Asai, Hiroyuki Shindo, Hideaki Takeda, and Yuji Matsumoto. 2020. LUKE: Deep Contextualized Entity Representations with Entity-aware Self-attention. In *Proceedings of the 2020 Conference on Empirical Methods in Natural Language Processing (EMNLP)*, pages 6442–6454, Online. Association for Computational Linguistics.

Lang Yu and Allyson Ettinger. 2020. Assessing Phrasal Representation and Composition in Transformers. In *Proceedings of the 2020 Conference on Empirical Methods in Natural Language Processing (EMNLP)*, pages 4896–4907, Online. Association for Computational Linguistics.

Mo Yu and Mark Dredze. 2015. Learning Composition Models for Phrase Embeddings. *Transactions of the Association for Computational Linguistics*, 3:227–242.

Zikai Zhou and Kei Wakabayashi. 2022. Topic Modeling using Jointly Fine-tuned BERT for Phrases and Sentences. In *The 14th Forum on Data Engineering and Information Management*.

# A Details on Training a Dedicated Model for Phrase Representation

## A.1 Triplet Loss

For training with the triplet loss, the anchor, positive and negative sentences are first extracted. For the anchor phrase, 100 sentences containing it are extracted from the corpus. The two furthest sentences, according to the embedding function used, are taken as the anchor and positive contexts. Then 300 random sentences that do not contain the phrase are extracted. The sentence closest to one of the two positive sentences are used as the negative context (with the phrase in that sentence as the negative phrase), and the closer positive sentence is the anchor sentence. The embedding function is the `McPhraSy` embedder, which is also trained and improved at the same time.

## A.2 Hyperparameters and Optimization

We train the `McPhraSy` embedding model with Adam optimizer and a learning rate of 7e-4. We set $m = 300$ for the number of contexts used (as examined in §5.2.2).

# B Dataset Preparation Details

## B.1 Collection of Phrase Groups

**Phrase extraction.** Noun phrases were extracted from reviews using the part-of-speech regular expression '`KT: {(<JJ>* <NN.*>+ <IN>)? <JJ>* <NN.*>+}`' of two to three words, and then lowercased for consistency. To save processing time, only the first 200 characters in each review were considered (an initial sampled process showed that this did not substantially change phrase frequencies).

Phrases were filtered out if they contained punctuation, personal pronouns (i, we, you, he, she, it, they, me, us, you, her, him, it, them, mine, ours, yours, hers, his, theirs, my, our, your, her, his, their, myself, yourself, herself, himself, itself, ourselves, yourselves, themselves, all, another, any, anybody, anyone, anything, both, each, either, everybody, everyone, everything, few, many, most, neither, nobody, none, no, one, nothing, one, other, others, several, some, somebody, someone, something, such), overly sentimental words (great, excellent, worst, best, good, nice, beautiful, bad, favorite, awesome, amazing, wonderful, quality, perfect, other, more, less, low, high, cute, pretty, adorable, ugly), or some substrings that render uninformative phrases (lot of, lots of, couple of, bit of).

**Phrase grouping.** Phrase grouping around a seed phrase was conducted with SpaCy similarity. If a phrase had a similarity above 0.99, it was not taken into the group.

## B.2 Initial Crowdsourcing Annotation Experiment

To cluster phrases within each of the groups, we first ran experimental crowdsourcing tasks, which we eventually dismissed. In the first task, a worker was shown a phrase from a group and the 24 other phrases in a separate list. The worker was to mark the phrase in the list that was most similar in meaning to the main phrase (or *None*). This task was conducted five times for each of the phrases in group (hence a pair of phrases in a group could be marked similar up to 10 times). Then, pairs that were marked together more than twice were used in the next annotation step.

In the second crowdsourcing task, a pair of phrases from a group was shown (those collected in the first step) with the 23 remaining phrases in a separate list. Here, a worker was to mark all the phrases that were similar to the pair (or *None* or *Pair is not similar*). The breakdown to two stages was performed to: (1) cut down on natural crowdsourcing noise due to unreliable annotations, and (2) to minimize the differences caused by subjective understanding of phrases. By presenting two phrases, workers would have a stronger anchor around which to find other similar phrases.

Even with this process, the final clusters formed were quite different, and it was unclear how to assemble the final clusters in an automatic manner, since our goal was to form a high quality dataset. The manual expert annotation labor required to aggregate the final clusters was not worth the crowdsourcing effort. After some attempts with a few groups, we resorted to internal expert annotation for clustering, as described in §4.1.

## B.3 Expert Annotation

**Annotation time.** Each group took an average of about 4.5 minutes to annotate, with time differences depending on the complexity of the group.

**Agreement.** On the 6 groups clustered by all three annotators, the average pairwise inter-annotator agreement scores were:

| | B | | | C | | |
|---|---|---|---|---|---|---|
| | **V** | **AR** | **AMI** | **V** | **AR** | **AMI** |
| **A** | 0.92 | 0.55 | 0.62 | 0.92 | 0.54 | 0.59 |
| **B** | | - | | 0.94 | 0.57 | 0.65 |

### B.4 Group Clustering Example

Table 4 presents an example of a group clustering.

## C Ethical Considerations

**Datasets.** The phrase similarity datasets (Turney and PPDB) were obtained in accordance to their license and terms of use.

**Crowdsourcing.** For the initial crowdsourcing experiments, we used the Appen[9] website, and employed workers from English speaking countries. We set a wage of $9/hour, according to a rate calculated by some internal testing of the tasks. Workers were given a qualification test before the assignments, consisting of example assignments from the actual task.

**Compute.** For all training and testing processes, we used a single NVIDIA 1080TI GPU. Training the `McPhraSy` embedding model takes about 1 hour. To cut down on processing time of each training experiment, we preprocessed all SpanBERT and Phrase-BERT representations once since they are frozen during training of our model (which took about 5 hours for the whole training set). Inferring one similarity for a pair of phrases is very quick, however computing the clusters of a group can accumulate to several minutes worth of pairwise similarities. We therefore kept a cache of similarities during our inference-time experiments for clustering, which significantly sped up the clustering procedure.

| |
|---|
| different sizes<br>variety of sizes<br>various sizes<br>multiple sizes |
| variety of projects<br>different projects |
| variety of ways<br>different ways |
| sturdy material<br>durable material<br>soft material<br>strong material<br>thin material<br>heavy material |
| variety of colors<br>assorted colors<br>color variety<br>different colours<br>different color |
| different brand |
| kinds of things |
| variety pack |
| different designs |
| make sure |
| different fabrics |

Table 4: A group of phrases, clustered to cases of *equivalent meaning* or *same of a kind* (the fourth cluster from the top). A cluster with one phrase is called a singleton cluster.

[9] https://appen.com/