No, you're not alone: A better way to find people with similar experiences on Reddit

Zhilin Wang **Elena Rastorgueva** **Weizhe Lin** **Xiaodong Wu**
University of Cambridge, United Kingdom
{zw322, wl356, xw338}@cam.ac.uk,
elenaras@cantab.net

Abstract

We present a probabilistic clustering algorithm that can help Reddit users to find posts that discuss experiences similar to their own. This model is built upon the BERT Next Sentence Prediction model and reduces the time complexity for clustering all posts in a corpus from $O(n^2)$ to $O(n)$ with respect to the number of posts. We demonstrate that such probabilistic clustering can yield a performance better than baseline clustering methods based on Latent Dirichlet Allocation (Blei et al., 2003) and Word2Vec (Mikolov et al., 2013). Furthermore, there is a high degree of coherence between our probabilistic clustering and the exhaustive comparison $O(n^2)$ algorithm in which the similarity between every pair of posts is found. This makes the use of the BERT Next Sentence Prediction model more practical for unsupervised clustering tasks due to the high runtime overhead of each BERT computation.

1 Introduction

On many subreddits within Reddit, such as r/Advice[1], users choose to share highly personal experiences that matter greatly in their lives in order to ask for advice from other users. Relative to other popular social networking sites such as Facebook, Instagram and Twitter, Reddit offers a greater extent of anonymity because there is no requirement for users to register accounts with their real names. Users are therefore often more at ease to reveal their experiences honestly and in full detail because the risk of facing repercussion from their real-life social networks is minimal. An example post is shown in Figure 1. This offers a unique opportunity to use what users posted on these subreddits as a proxy for their real-life experiences,

Title: "How do I get Vaccinated as a Minor?"
Body: "I am a 16 year old female whose mother became anti-vax a couple of years ago when she got Facebook. I don't think I've gotten a vaccine in 4-6 years at this point.I really want to get vaccinated ..."

Figure 1: An excerpt from a /r/Advice subreddit post.

and what they felt and thought about these experiences. Here, we attempt to cluster similar posts on these subreddits. In this aspect, we are not only interested in the circumstances under which the individuals encountered the experiences, but also how they responded to the various situations in terms of their actions, thoughts and feelings.

To do so, we take advantage of recent improvements in transformer-based mechanisms for transfer learning, most prominently BERT (Devlin et al., 2019). This capacity allows a model to be pre-trained on a large corpus unrelated to our specific task, in order to learn fundamental statistical properties of language relating to syntax and semantics. Specifically, we employ the model of BERT that is pre-trained for the task of Next Sentence Prediction, which seeks to capture semantic congruence between two paragraphs. While this is not entirely similar with finding semantic similarities within Reddit posts (which often contain informal language), we hypothesize that some information encapsulated in the pre-trained model will be transferable to our task. In this way, we can train a model using an extremely limited dataset of around 9000 posts. Currently, we have only trained our model using an English corpus, but given that the BERT model (Devlin et al., 2019) has multi-lingual capabilities, we believe that our findings can apply to languages other than English.

Our key contribution lies in clustering all posts

[1] https://www.reddit.com/r/Advice/

Proceedings of the 2019 EMNLP Workshop W-NUT: The 5th Workshop on Noisy User-generated Text, pages 307–315
Hong Kong, Nov 4, 2019.

into groups without needing to calculate the pairwise Next Sentence Prediction likelihood for every pair of posts. This reduces the computational complexity of this process from $O(n^2)$ to $O(nm)$, where n is the number of posts, m is the number of clusters and $n \gg m$. This is an important advancement because an operation on each pair of posts is in itself computationally intensive due to the transformer architecture. Our design can enable such clustering to be more scalable for larger corpora.

2 Related work

2.1 Deductive coding of individual experiences

The first field of related work lies in attempts to create a standard for deductive coding of individual experiences, typically based in the field of psychology. In this approach, trained individuals inspect people's writing of their experience and classify each into a predefined category. The inspection of each individual is then compared to that of others to ensure consistency. Demorest et al. (1999) defined an individual experience in terms of a person's wish, a response from another person and a response from the self in light of the other person's response. At each stage, experiences can be classified into categories such as wanting to be respected, being disliked by others and feeling disappointed because of the rejection. On the other hand, Thorne and McLean (2001) defined experiences in terms of themes. These themes include occasions of life threatening events, relationship-related issues and a sense of mastery. Together, these can provide a basis for identifying the elements within a post that can be used to compare to other posts.

2.2 Computational personality differences

The second field of related work lies in research on how individuals differ in terms of their responses to common life situations and how such differences can be measured by analyzing their writing. The most popular measure is the Myers-Briggs Type Indicator (Myers et al., 1990; Gjurković and Šnajder, 2018), which classifies individuals into 16 types based on their disposition. Another common measure is the Big Five personality traits (Yarkoni, 2010), which gauges people in terms of five dimensions: Openness, Conscientiousness, Extraversion, Agreeableness and Neuroticism. Pennebaker (2011) also investigated how other personality attributes such as a focus on the self (as opposed to others) and differences in status can be predicted based on word choices. All of the above measures seek to group people into distinct categories based on how they write. The relative success of this field in doing so convinced us that it is possible to capture individual differences through a person's writing. However, to help Reddit users find other users with similar experiences, we are interested in not only the general response patterns of an individual but also their specific response to a specific situation. This means that we cannot directly adopt their methodology of performing a supervised classification task. Instead, we decided on unsupervised methodology because it would permit a wider range of situations and responses.

2.3 Analysis of characters and plots in novels and movies

The final field of related work comes from the computational analysis of characters and plots in novels and movies. Bamman et al. (2013, 2014) sought to classify characters into various prototypes in film and novels. Frermann and Szarvas (2017) and Iyyer et al. (2016) went a step further to classify the types of relationships that exist between main characters in a novel, in addition to the prototype of each character in novels. These works inspired this paper on Reddit posts, because events in many novels and movies are relatable to the experiences of real-life individuals. Furthermore, many posts also concern interactions between the author and other people in the author's real-life social networks. However, Reddit posts are much shorter than movies and novels. This means that the recurrent models designed to represent how a character/relationship develops through a novel/movie in literature above is less applicable in our research. Moreover, unlike novels/-movies, which often use character names together with personal pronouns, Reddit posts tend to use personal pronouns almost exclusively (in order to preserve anonymity). As a result, a popular coreference resolution framework[2] would not work on Reddit posts. Therefore, most of the methods described in literature above could not be adapted for our research and we had to look elsewhere for a suitable architecture.

[2] https://github.com/huggingface/neuralcoref

Algorithm 1

```
procedure ONE-PROB-CLUSTERING(posts, m)
    unselected_posts ← posts; clusters ← {}
    while unselected_posts ≠ [ ] do
        selected_post ← RANDOM-SELECTION(all_unselected_posts)        ▷ Without replacement
        for query_post in all_unselected_posts do
            similarity ←BERT-NEXT-SENTENCE-PREDICTION(selected_post.title, query_post.text)
        most_similar ← most similar ⌊n/m⌋ query posts
        clusters [selected_post] ← most_similar
        for post in most_similar do
            unselected_posts.remove(post)
    return clusters
```

3 Probabilistic Clustering

3.1 Data preprocessing

We downloaded 200 days of posts from the r/Advice subreddit[1] using the Pushshift API[3]. After that, we filtered out posts with (i) scores lower than 3 based on the number of upvotes, downvotes and comments they received, which indicated that they might not be pertinent to the users of the subreddit, and (ii) no textual information in the post. This left us with 8865 posts.

3.2 Generating similarities between two posts

We then used the BERT Next Sentence Prediction model[4] to predict the likelihood that the title from post A will be proceeded by the body text of post B. The model had been pre-trained on the BooksCorpus (800M words) (Zhu et al., 2015) and English Wikipedia corpus (2,500M words). During the pre-training process, half of the inputs consist of sentence B being the actual sentence following sentence A (labeled as 'IsNext') while the other half consists of a random sentence from the corpus that does not proceed sentence A (labeled as 'NotNext') (Devlin et al., 2019). We found this to be a feasible method of deciphering the semantic similarity between the title of post A and the body text of post B because in more than 97.7% of our posts, the text is predicted to follow its own title. This is likely because the pre-training task of finding sentences that are likely to follow one another is highly similar to our task of finding text of a post that is likely to be after the title of the same post. While the BERT Next-Sentence-Prediction model was pre-trained on a sentence-level corpus, this result demonstrates that its effects can translate to a paragraph-level task, which was also noted by Devlin et al. (2019).

Besides using the title of post A and the text of post B, we also experimented with the title of post A and the title of post B as well as the text of post A and the text of post B.

3.3 Clustering based on similarities

Intuitively, clustering can be done by comparing each post with all other posts in the corpus. This would be a $O(n^2)$ operation where n is number of posts. However, due to the large number of weights of the BERT model, each comparison takes a long time to complete. Therefore, even for a small corpus of 8865 posts, it would be infeasible to perform pairwise comparison of every pair. This makes the intuitive algorithm highly unscalable with the number of posts.

To overcome this problem, we invented a probabilistic clustering architecture, described in Algorithm 1.

The computational complexity of this algorithm, ONE-PROBABILISTIC-CLUSTERING can be calculated as follows. Given that the most runtime-intensive step is BERT-NEXT-SENTENCE-PREDICTION, we can choose to solely focus our analysis on this step.

In each while-loop, we have to perform the BERT-NEXT-SENTENCE-PREDICTION process $n_{unselected\ posts}$ times. $n_{unselected\ posts}$ starts from n and decreases by $\lfloor n/m \rfloor$ after each while-loop, where n is the number of posts and m is the number of clusters.

Therefore, S, the total number of times the BERT-NEXT-SENTENCE-PREDICTION process is carried out, follows an arithmetic progres-

[3] https://github.com/dmarx/psaw

[4] The uncased small model on https://github.com/huggingface/pytorch-transformers

Algorithm 2

```
procedure MERGE-MULTIPLE-PROB-CLUSTERING(p, m, posts)
    similarity_table ← {}
    for i ← 0, n − 1 do
        similarity_table[i] = [0, 0, ..., 0, 0]          ▷ Initialise with array of size n
    for i ← 0, p − 1 do
        one_probabilistic_clustering = ONE-PROBABILISTIC-CLUSTERING(posts, m)
        for j in one_probabilistic_clustering.keys() do
            One_cluster = [j] + [one_probabilistic_clustering[j]]
            all_similar_pairs = PERMUTATIONS(one_cluster, 2)
            for k in all_similar_pairs do
                similarity_table[k[0]][k[1]] += 1
    return similarity_table
```

Algorithm 3

```
procedure GENERATE-CLUSTERS-FROM-SIMILARITY(similarity_table, m, n)
    unselected_posts ← posts; clusters ← {}
    while unselected_posts ≠ [ ] do
        selected_post = RANDOM-SELECTION(unselected_posts)          ▷ Without replacement
        sort similarity_table[selected_post]
        most_similar = most similar ⌊n/m⌋ posts in unselected_posts
        clusters [selected_post] = most_similar
        for post in most_similar do unselected_posts.remove(post)
    return clusters
```

sion:

$$S = n + (n - \lfloor n/m \rfloor) + (n - 2\lfloor n/m \rfloor) + \\ \cdots \\ (n - (m-1) * \lfloor n/m \rfloor) + (n - m * \lfloor n/m \rfloor) \\ \leq \frac{n * m}{2} \tag{1}$$

Therefore the time complexity of ONE-PROBABLISTIC-CLUSTERING is $O(nm)$. We chose $m = 30$ because initial experiments using a Gaussian Mixture Model to cluster BERT sentence embedding of Reddit post text (by average-pooling all tokens in the second-to-last layer)[5] suggested that $m = 30$ is the optimal choice because it scored lowest on the Akaike Information Criteria (Akaike, 1973). The absolute computational complexity for ONE-PROBABLISTIC-CLUSTERING, taking into consideration the cost of sorting most_similar is $O(mnlogn)$. When n is small however, the constant factor for BERT-NEXT-SENTENCE-PREDICTION is so great that it dominates the run-time, allowing the run-time to $O(nm)$.

[5] https://github.com/hanxiao/bert-as-service

The time complexity for Algorithm 2, MERGE-MULTIPLE-PROB-CLUSTERING, is $O(n^2)$ where n is the number of posts used to generate an n-by-n matrix for the similarity table. However, because the constant factor is so large, when $n = 8865$, it is the time complexity from running ONE-PROBABILISTIC-CLUSTERING that dominates. Therefore, the runtime complexity of MERGE-MULTIPLE-PROB-CLUSTERING is $o(nmp)$, $n = 8865$ where n is the number of posts, m is the number of clusters and p is the number of times that ONE-PROBABILISTIC-CLUSTERING is run. A value of $p = 5$ is chosen because although a more informative similarity table will be constructed when p is higher, it also requires more computational resources.

GENERATE-CLUSTERS-FROM-SIMILARITY has a time complexity $O(mn \log n)$ where n is the number of posts and m is the number of clusters because the while-loop will run for m iterations with each iteration taking $O(n \log n)$ for sorting. In practice however, the runtime is dominated by the previous MERGE-MULTIPLE-PROBALISTIC-CLUSTERING step due to its large constant factor.

3.4 Fine-tuning BERT Next Sentence Prediction

Besides using the pre-trained Next Sentence Prediction model, we also fine-tuned the model using posts from Reddit to better fit the classification to our corpus. We used not only 8865 posts from r/Advice, but also over 300,000 posts from similar subreddits[6]. During the pre-training process, we focused on training the weights for the final BERT pooling layer as well as the classification layer and froze the parameters in all other BERT layers. We made this decision because our corpus was not sufficient for us to retrain the parameters for the layers beneath and doing so might lead to worse performance than using the default pre-trained parameters. Because fine-tuning requires labeled data, we performed fine-tuning based on posts from the same author. In the subreddits that we used, some authors posted multiple times to share about a similar experience. This is likely because they did not receive adequate guidance from the Reddit community after their earlier post(s). Therefore, two posts from the same subreddit and the same author are likely to discuss about similar themes and topics. We used this tendency to generate text-text pairs from the same author with a label 'IsNext' and paired one text with a randomly selected text from another post not from the same author with a label 'NotNext'.

4 Evaluation

4.1 Qualitative Evaluation

Table 1, shows the titles of 3 randomly chosen posts and the five most similar posts to them. There is a high degree of coherence, and the posts are not only similar thematically (in Post 1: pregnant - pregnancy test - dating - hookups), but also emotionally (in Post 2: a sense of succorance) and at a word-level (in Post 3: 'dog').

4.2 Quantitative Evaluation

Baselines

Baseline measurements were done using Latent Dirichlet Allocation (LDA) (Blei et al., 2003) and word2vec (Mikolov et al., 2013). LDA document-topic mappings were performed using Gensim[7]. Documents were first tokenized, removed of stopwords and lemmatized. A Bag of Words (BoW) corpus was obtained before a term frequency-inverse document frequency (TF-IDF) corpus was derived from it. Topic modeling was then performed on both the BoW corpus (thereafter LDA-BoW) and TF-IDF corpus (thereafter LDA-TFIDF) with the number of topics set to 30, in line with the number of clusters used. The document-topic mapping of each post is then used for computing cosine similarities with all other posts.

Word2Vec embeddings were also used as a benchmark. Specifically, pre-trained word2vec embeddings of dimension 300 (Mikolov et al., 2013) were used to generate two forms of sentence embeddings. The first (thereafter called W2V-Weighted) is calculated by weighing the contribution of each word embedding by the inverse of its relative frequency to the final sentence embedding. In doing so, the contributions of the most common words are minimized. The second (thereafter called W2V-SIF) is calculated by first taking the weighed sentence embedding before removing the first principal component from it. (Arora et al., 2017).

Generating similarities from baselines

Cosine similarities were then calculated between all documents. The resulting cosine similarity matrix was be then entered in the GENERATE-CLUSTERS-FROM-SIMILARITY function (Algorithm 3) with the number of clusters (m) and number of posts (n) kept the same as in the probabilistic clustering model.

Evaluation metrics

To determine if our clustering algorithm is better than baselines, it is imperative to have evaluation metrics. However, because our clustering tasks do not have ground truth labels, we could not find common metrics to evaluate the effectiveness of our algorithm. Therefore, we designed two novel extrinsic metrics for this purpose.

Evaluation Metric 1: Same author score

We designed this metric based on the observation that authors who post multiple times in the r/Advice subreddit tend to post about similar topics. Therefore, a good clustering algorithm might be more effective at clustering posts from the same author in the same cluster. To measure this, we found all pairs of posts with the same author and

[6] r/depression, r/relationship_advice, r/offmychest, r/IAmA, r/needadvice, r/tifu, r/confessions, r/confession, r/TrueOffMyChest, r/confidence, r/socialanxiety, r/Anxiety, r/socialskills.

[7] https://github.com/RaRe-Technologies/gensim

Randomly chosen post titles Titles of 5 most similar posts
Post 1: I'm afraid I could be pregnant
1. I bought pregnancy tests bc im paranoidddd
2. What should I think of this convo between my ex and I? I felt guilty...
3. Met an E-guy became really good friends, started 'dating'. We joked sexually and I found it funny.
4. My entire life I skipped school and just didn't really care too much for it.
5. I'm scared of hookups.
Post 2: How do I help support my girlfriend who has been raped?
1. Should this teacher get in trouble for making these comments about male students?
2. How do I help my (severely?) mentally ill daughter?
3. I've been avoiding my family for the past few months, I'm not sure what to do now.
4. How to turn you life around after doing almost nothing for 3-4 years.
5. An old tradition brought me into a rather messed up situation.
Post 3: Neighbor is going nuts and wants to shoot my dog (or my sister, or all of us...)
1. How long do I wait before calling the police for a welfare check?
2. How to help my roommate - crying in his sleep.
3. To dog or not to dog?
4. All day every day my neighbors dog is on a 6-10 ft rope.
5. Dog sitting disaster

Table 1: Each unshaded box shows the titles of a randomly chosen post and the 5 posts most similar to it

counted the proportion of them that are clustered into the same cluster. Finally we account for the likelihood that they were arranged into the same cluster by chance, which is a constant equal to $\frac{1}{m}$. This is described in Eq. 2, where I is the total number of authors and j represents possible combinations of pairs of posts by a single author (hence there are a total of J_i possible combinations for author i). j^0 and j^1 represent the first and second post in the pair, respectively, and e.g. $Cluster(j^0)$ returns the i.d. of the cluster that j^0 has been assigned to.

$$S_{same\,author} = \frac{\sum_{i=1}^{I}\sum_{j=1}^{J_i} \mathbb{1}_{Cluster(j^0)=Cluster(j^1)}}{\sum_{i=1}^{I}\sum_{j=1}^{J} 1} - \frac{1}{m} \tag{2}$$

Evaluation Metric 2: Jaccard score This metric was inspired by the observation that authors who share similar interests tend to post about similar topics on r/Advice. In this case, we measure how similar the interests of the authors are by counting the number of subreddits they have both posted and commented on divided by the number of the union of subreddits they have posted and commented on. In the case that they have both commented and posted in exactly the same set of subreddits, their Jaccard scores will be 1. If they have not posted or commented on any subreddits in common, their Jaccard score will be 0. In our use case, however, the lower bound is strictly higher than 0 because two authors would have both posted in r/Advice. Furthermore, to prevent over-accounting for throwaway accounts, which are only used to post once, we set the Jaccard score of any pair of posts, which consists of at least one that has only posted/commented in one subreddit to 0.

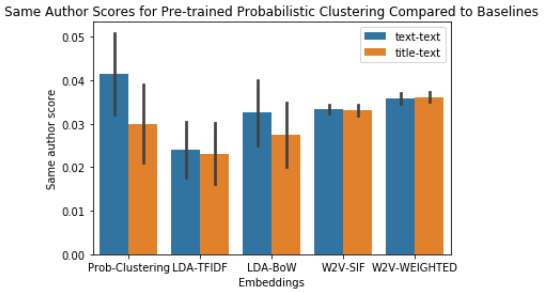
Jaccard scores of all pairs of posts in the same cluster were then added together. The result was then multiplied by the number of clusters to account for the difference in probability that two posts will be put into the same cluster by chance alone, which varies inversely with the number of clusters. Finally, the result is divided by the square of the number of posts because the number of posts in each cluster varies linearly with the total number of posts and therefore the number of combination of two posts in the same category varies linearly to the square of the number of posts.

The metric is described in Eq. 3 where I is the total number of clusters (referred to as m in the rest of this paper) and J_i represents all possible combinations of pairs of posts in the same cluster.

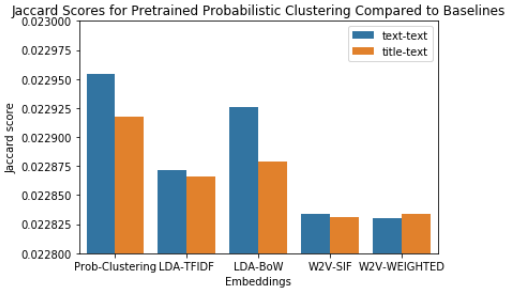
$L_{j^0}^{posted}$ represents the set of subreddits that the author of the first post in the pair has posted to. Likewise j_1 refers to the second author and ^{commented} refers to subreddits the author commented on.

$$S_{Jaccard} = \sum_{i=1}^{I}\sum_{j=1}^{J_i} 0.5\left(\frac{L_{j^0}^{posted} \cap L_{j^1}^{posted}}{L_{j^0}^{posted} \cup L_{j^1}^{posted}} + \frac{L_{j^0}^{commented} \cap L_{j^1}^{commented}}{L_{j^0}^{commented} \cup L_{j^1}^{commented}}\right) * \frac{n_{clusters}}{n_{posts}^2} \quad (3)$$

4.3 Effectiveness of probabilistic clustering



(a)

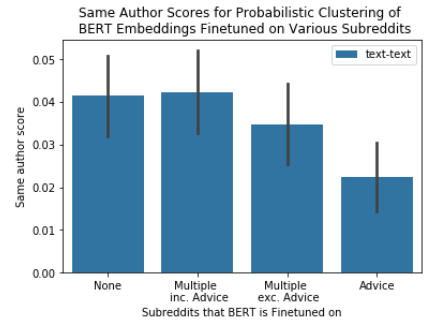


(b)

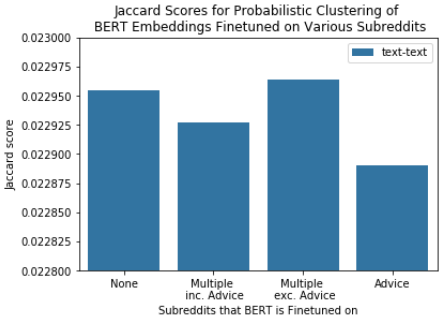
Figure 2: Same author (a) and Jaccard scores (b) for pre-trained probabilistic clustering compared to baselines. Scores generated by performing 100 iterations and finding the average. Error bars in (a) represent the standard deviation. Error bars are not shown on (b) because standard deviation is insignificant relative to the scale of the figure. Higher is better on both figures.

Figure 2 shows that probabilistic clustering performs better than all baseline embeddings. This is due to the model's capability to learn complex relationships between the two input sentences instead of using cosine distance as the measure of similarity. In nearly all embeddings, using the body text from two posts surpasses the performance of using the title from one post and the body text from the second post, measured in both metrics. This is likely because the body text typically contains more words, which can provide more information for sentence embeddings. A review of the topic-word mapping for LDA suggests that words were not clearly resolved into distinct topics. Words like "friends", "dating" and "fun" appeared in nearly half of all topics, suggesting that LDA might be inadequate for capturing topics in such Reddit posts. This might be because Reddit posts are mostly informal and uses a limited range of vocabulary with many common words used in different context with highly distinct meanings. Using a transformer-based BERT architecture might be better able to capture such contextual information (Vaswani et al., 2017).

4.4 Fine-tuning



(a)



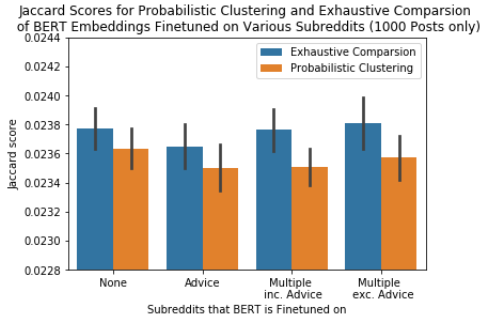
(b)

Figure 3: Same author (a) and Jaccard scores (b) for probabilistic clustering of BERT embeddings fine-tuned on subreddit(s) based on the same author selection criteria. Scores generated by performing 100 iterations and finding the average, with error bars in (a) representing the standard deviation. Error bars not shown on (b) because standard deviation is insignificant relative to the scale of the figure. Higher is better on both figures.

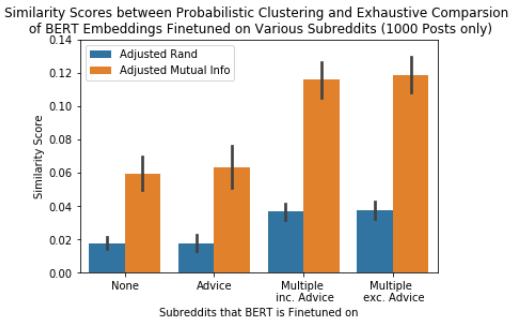
Figure 3b suggests that while some forms of fine-tuning can perform better than the pre-trained model, improvements are usually modest and inconsistent. In both Figures 3a and 3b, the model fine-tuned using only the Advice subreddit performed worst in terms of both metrics suggesting that pre-training on a highly limited corpus should be avoided. Furthermore, its poor performance

also supports the hypothesis that the model does not simply memorize the seen corpus because the same set of the corpus was used to train and to test the classifier. On the contrary, pre-training the model with a large corpus, even one that does not contain the testing sample, can lead to some improvement in the Jaccard score, as in Figure 3b. This suggests a lack of over-fitting in our model and correspondingly indicates a high possibility of generalizing our model beyond our corpus.

4.5 Ablation experiments



(a)



(b)

Figure 4: (a) Jaccard Scores for the first 1000 posts, obtained from Probabilistic Clustering and Exhaustive Comparison. (b) Similarity scores between Probabilistic clustering and Exhaustive Comparison for the first 1000 posts. Scores generated by performing 50 iterations and finding the average. Error bars represent standard deviation. Higher is better on both figures.[8]

An ablation study was conducted to investigate the contribution of the probabilistic clustering algorithm beyond the value of the BERT Next Sentence Prediction task. In the exhaustive comparison control, the likelihood that the post text of post A will follow the post text of post B was found for all combinations of pairs of posts. This allowed us to construct a complete similarity table that can be used to generate clusters using the procedure GENERATE-CLUSTERS-FROM-SIMILARITY. The same parameters of 1,000 posts and 30 clusters were chosen for both probabilistic clustering and the exhaustive comparison. Figure 4a suggests that while exhaustive comparison led to higher performance, the difference is typically less than a standard deviation. This means that the reduction in performance is relatively minimal compared to the significant reduction in time complexity, which might make exhaustive comparison increasingly unfeasible as the number of posts increases.

In Figure 4b, Adjusted Rand score (Hubert and Arabie, 1985) and Adjusted Mutual Information (Vinh et al., 2010) were used to measure for coherence between exhaustive comparison and probabilistic clustering. Both scores suggest that the probabilistic clustering of embeddings pre-trained on all corpuses agree with the exhaustive comparison to a degree that is significantly higher than chance would predict (when both scores would be 0). Furthermore, the model pre-trained on the larger corpora of (i) Multiple subreddits inc. Advice and (ii) exc. Advice agree more with the exhaustive comparison. This might be because the models fine-tuned on those corpora are better able to understand the use of language used on Reddit and hence better able to accurately choose the top $\lfloor n/m \rfloor$ similar posts. However, this does not translate to higher Jaccard scores compared to the pre-trained model. This could be due to (i) the Jaccard score metric (measuring the proportion of subreddits that two authors have posted or commented on in common) not being able to fully capture all information that explain the similarity between posts, or (ii) the BERT Next Sentence Prediction model (pre-trained and minimally fine-tuned) being unable to fully capture the relative similarities between posts, even though it is capable of capturing absolute similarities between the title of a post and its post text.

5 Conclusion

In conclusion, we have presented a probabilistic clustering algorithm for clustering similar posts on many subreddits on Reddit such as r/Advice. This algorithm is built on top of the BERT Next Sentence Prediction model and reduces the time complexity of clustering posts from $O(n^2)$ to $O(n)$ with respect to the number of posts. This algorithm can be helpful for users on Reddits to find

[8] Only Jaccard scores were used to compare the extrinsic performance of both methods because the low number of pairs of post that share the same author makes the same author score uninformative.

posts similar to those they have written themselves (about their own experiences) or others that they are interested in. To further build on the contribution of this work, we encourage researchers to experiment with alternative fine-tuning methods as well as performing post-processing of similarity tables such as performing post-level normalization to reduce the occurrence of some posts being highly similar to a great number of posts while others being similar to too few. Researchers may also consider incorporating other textual information such as comments on Reddit posts into the model to improve its performance.

References

Hirotugu Akaike. 1973. Information theory and an extension of the maximum likelihood principle. *Czaki, Akademiai Kiado, Budapest*.

Sanjeev Arora, Yingyu Liang, and Tengyu Ma. 2017. A simple but tough-to-beat baseline for sentence embeddings. In *ICLR*.

David Bamman, Brendan O'Connor, and Noah A. Smith. 2013. Learning latent personas of film characters. In *Proceedings of the 51st Annual Meeting of the Association for Computational Linguistics (Volume 1: Long Papers)*, pages 352–361, Sofia, Bulgaria. Association for Computational Linguistics.

David Bamman, Ted Underwood, and Noah A. Smith. 2014. A Bayesian mixed effects model of literary character. In *Proceedings of the 52nd Annual Meeting of the Association for Computational Linguistics (Volume 1: Long Papers)*, pages 370–379, Baltimore, Maryland. Association for Computational Linguistics.

David M. Blei, Andrew Y. Ng, and Michael I. Jordan. 2003. Latent dirichlet allocation. *J. Mach. Learn. Res.*, 3:993–1022.

Amy Demorest, Paul Crits-Christoph, Mary Hatch, and Lester Luborsky. 1999. A comparison of interpersonal scripts in clinically depressed versus nondepressed individuals. *Journal of Research in Personality*, 33(3):265 – 280.

Jacob Devlin, Ming-Wei Chang, Kenton Lee, and Kristina Toutanova. 2019. Bert: Pre-training of deep bidirectional transformers for language understanding. In *NAACL-HLT*.

Lea Frermann and György Szarvas. 2017. Inducing semantic micro-clusters from deep multi-view representations of novels. In *Proceedings of the 2017 Conference on Empirical Methods in Natural Language Processing*, pages 1873–1883, Copenhagen, Denmark. Association for Computational Linguistics.

Matej Gjurković and Jan Šnajder. 2018. Reddit: A gold mine for personality prediction. In *Proceedings of the Second Workshop on Computational Modeling of People's Opinions, Personality, and Emotions in Social Media*, pages 87–97, New Orleans, Louisiana, USA. Association for Computational Linguistics.

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

Mohit Iyyer, Anupam Guha, Snigdha Chaturvedi, Jordan Boyd-Graber, and Hal Daumé III. 2016. Feuding families and former Friends: Unsupervised learning for dynamic fictional relationships. In *Proceedings of the 2016 Conference of the North American Chapter of the Association for Computational Linguistics: Human Language Technologies*, pages 1534–1544, San Diego, California. Association for Computational Linguistics.

Tomas Mikolov, Ilya Sutskever, Kai Chen, Greg S Corrado, and Jeff Dean. 2013. Distributed representations of words and phrases and their compositionality. In C. J. C. Burges, L. Bottou, M. Welling, Z. Ghahramani, and K. Q. Weinberger, editors, *Advances in Neural Information Processing Systems 26*, pages 3111–3119. Curran Associates, Inc.

Isabel Briggs Myers, Mary H.McCaulley, and Allen L. Hammer. 1990. *Introduction to Type: a description of the theory and applications of the Myers-Briggs type indicator*. Consulting Psychologists Press.

James W. Pennebaker. 2011. *The secret life of pronouns: What our words say about us*. Bloomsbury Press, New York, NY.

Avril Thorne and Kate C. McLean. 2001. Manual for coding events in self-defining memories. Unpublished Manuscript.

Ashish Vaswani, Noam Shazeer, Niki Parmar, Jakob Uszkoreit, Llion Jones, Aidan N. Gomez, Lukasz Kaiser, and Illia Polosukhin. 2017. Attention is all you need. In *NIPS*.

Nguyen Xuan Vinh, Julien Epps, and James Bailey. 2010. Information theoretic measures for clusterings comparison: Variants, properties, normalization and correction for chance. *J. Mach. Learn. Res.*, 11:2837–2854.

Tal Yarkoni. 2010. Personality in 100,000 words: A large-scale analysis of personality and word use among bloggers. *Journal of research in personality*, 44(3):363373.

Yukun Zhu, Jamie Ryan Kiros, Richard S. Zemel, Ruslan Salakhutdinov, Raquel Urtasun, Antonio Torralba, and Sanja Fidler. 2015. Aligning books and movies: Towards story-like visual explanations by watching movies and reading books. *2015 IEEE International Conference on Computer Vision (ICCV)*, pages 19–27.