Research Paper Tagger

Syed Arbaaz Qureshi {squreshi

Avani Jindal avanijindal

Muskan Jain muskanjain

Sahil Jindal sahiljindal}

1 Problem statement

Oftentimes, while submitting a research article to a conference, we are asked to tag the research tracks of our article. Students and novice researchers find it difficult to tag the appropriate tracks because of inexperience. To simplify this process, we propose to build a model named Research Paper Tagger (**RPT**) which automatically suggests tracks for the research papers, by looking at a combination of its title, abstract, and the names of its authors. We evaluate **RPT** on the Association for Computational Linguistics (ACL) conference. Realization of **RPT** is helpful in another way too. Every year, thousands of papers are submitted to various research conferences. The number of papers submitted have been on the rise in recent years, especially in AI conferences (3350 papers were submitted to ACL 2020). With these many papers being submitted, it becomes difficult to manually read the abstracts and assign the right reviewers. Conferences assign the appropriate reviewers based on the research tracks chosen by the submitter. **RPT** could be used for verifying these tagged tracks, specially in the case where the papers have been submitted and tagged by novice researchers.

We build an **RPT** specifically for ACL papers. To the best of our knowledge, there exists no dataset of ACL papers, with their corresponding research tracks. So we create a new dataset by extracting data from the ACL website [1]. Our dataset has ACL papers of the years 2019, 2020 and 2021, with their corresponding research track labels. You can find the code and the dataset at https://github.com/arbaazQureshi/research_paper_tagger

2 What we proposed vs. what we accomplished

In our project proposal we introduced a flow of our project. Our proposal had the following tasks (completed tasks have been struck out):

- ~~Scrape the research papers from ACL website and parse them to get the title, authors name and abstract~~. We didn't collect the introduction due to a few reasons, explained in the section 4.
- ~~Build and train TF-IDF (Baseline$_{\text{TF-IDF}}$) and RNN (Baseline$_{\text{RNN}}$) models on collected dataset and examine their performance~~
- ~~Use pre-trained BERT model and finetune it on our dataset. Examine its performance.~~
- ~~Perform in-depth error analysis to figure out what kinds of examples our approach struggles with~~

3 Related work

Unsupervised topic modeling: Unsupervised topic modeling is a well studied problem. We expound on a few standard unsupervised topic modeling approaches here. (Trstenjak et al., 2014) introduces the idea of using a TF-IDF (Term Frequency-Inverse Document Frequency) (Sammut and Webb, 2010) and KNN (K-Nearest Neighbour) module to classify text data. (SW. Kim, 2019) use Latent Dirichlet Allocation (LDA) (Blei et al., 2003) to extract keywords from the abstracts of various research papers. They use K-means clustering based on TF-IDF values to classify the papers with similar subjects under various categories. (Korlapati et al., 2019) compare LDA, Naive Bayes and Support Vector Machines (Cortes and Vapnik, 1995) for text classification. They use the Cora dataset (A.K. et al.,

[1] https://virtual.acl2020.org/papers.html?filter=titles&track=All+tracks

2000) and observe that LDA performs better and is more efficient for topic modelling. They show the trends observed from various conferences across three topics, relations between them, influence of various topics in those fields, and share insights about future topic prediction. (Kandimalla et al., 2021) use a slightly different approach. They use a deep attentive neural network which consists of two bi-directional recurrent neural networks followed by an attention layer (Vaswani et al., 2017). This network is trained on nine million abstracts from Web of Science. They classify the papers on the basis of their abstracts and conclude that retraining word embedding models is important and it enhances vocabulary overlap and improves the efficiency of attention mechanism. (Abuhay et al., 2018) employ the ICCS (International Conference on Computational Science) corpus which has 5982 papers collected over the span of 17 years. Both LDA and Non-negative Matrix Factorization (NMF) were used on it for topic modelling and it was observed that NMF performed better on distinct topics. They use trend analysis and Change Point Analysis (CPA) to identify the trends of topics which are appearing and disappearing. (Bafna et al., 2016) use TF-IDF and hierarchical agglomerative clustering and fuzzy K-means clustering algorithm. They calculate entropy to verify the quality of clusters. To estimate the number of clusters, they use silhouette coefficient. This work can be extended further and applied to get semantic relations which are domain-specific.

Supervised topic modeling: (Mcauliffe and Blei, 2008) propose a method called supervised Latent Dirichlet Analysis (sLDA). It is a statistical model of labelled documents The authors derive a maximum-likelihood procedure for parameter estimation, which relies on variational approximations to handle intractable posterior expectations. They test sLDA on two real-world problems: movie ratings predicted from reviews, and web page popularity predicted from text descriptions. (Lenc and Král, 2018) focus on automatic multi-label document classification of Czech text documents. They use deep neural networks that learn from simple features. Two different networks are compared: the first one is a standard multi-layer perceptron, while the second one is a popular convolutional network. The experiments on a Czech newspaper corpus show that both networks significantly outperform baseline method which uses a rich set of features with maximum entropy classifier. We have also shown that convolutional network gives the best results.

(Devlin et al., 2019) argued how the then existing standard language models, being unidirectional limited the choice of architectures that can be used for pretraining and proposed an improved fine-tune based approach BERT. While (Liu et al., 2019) explores the use of BERT model in combination with the Bayesian network to achieve a text classification task, (Yu et al., 2021) proposes the use of BERT-BiGRU model structure for classifying. In the latter, BiGRU model is specifically implemented to extract the text information features from both directions at the same time. Further, (Mosbach et al., 2021) gave a better understanding on how we can address the optimization difficulties and improve BERT's stability.

4 Data

We collect the research papers and their research tracks from the Association for Computational Linguistics (ACL) conference. We take the long and short papers of 2019, 2020 and 2021, and extract the title, abstract and the author list of the papers. In order to get the labels, we refer to the 'conference program' section of the proceedings, which lists the paper titles and the sessions in which they are presented. The session names are of the form 'Session 1A: Computational Social Science and Cultural Analytics 1', or 'Session 3C: Machine Translation and Multilinguality 3', where each name had the session title ('Session 1A', or 'Session 3C'), and a number at the end, indicating the number of times that particular session has been held before in the conference. We remove these two fields to get the label for the papers in the session (the papers presented in Session 1A: Computational Social Science and Cultural Analytics 1 would be labeled with 'Computational Social Science and Cultural Analytics'). We then group all the sessions that have the same label. This way, we get all the paper titles and their labels of one ACL conference. We do this for the ACL conferences of 2019, 2020 and 2021, and collect about 1744 paper titles and their labels.

We extract the author names and the abstract of the papers from the ACL website. We don't design a web scraper for doing this, we just copy-paste the title-abstract-authors from the ACL anthology

Research track	No of papers
Machine Translation and Multilinguality	187
Information Extraction, Retrieval and Text Mining	171
Semantics	167
Machine Learning for NLP	166
Dialogue and Interactive Systems	146
Generation	108
NLP Applications	106
Question Answering	100
Resources and Evaluation	93
Sentiment Analysis, Stylistic Analysis, and Argument Mining	78
Summarization	65
Computational Social Science, Social Media and Cultural Analytics	64
Language Grounding to Vision, Robotics and beyond	56
Tagging, Chunking, Syntax and Parsing	51
Interpretability and Analysis of Models for NLP	50
Linguistic Theories, Cognitive Modeling and Psycholinguistics	39
Discourse and Pragmatics	31
Phonology, Morphology and Word Segmentation	25
Speech and Multimodality	22
Ethics in NLP	19

Table 1: Class distribution.

website, and parse the data dump to get the three fields.

Once we have the raw research track labels and their associated papers, we merge a few pairs/triplets of research track labels. For example, three of the raw labels were 'Sentence Level Semantics', 'Word Level Semantics' and 'Lexical Semantics'. We merge these three labels into a single label 'Semantics', as the three raw labels contain similar papers which talk about semantics. Also, the number of papers under each of the three raw labels were small. After this merging of a few raw labels, we are left with 20 final research track labels. The labels, along with the class distribution is shown in the table 1. Notice that the tracks are fairly distinct; no two tracks have papers which have described similar works. We therefore formulate the problem at hand as a multi-class classification (instead of multi-label classification, as was proposed in the proposal).

In total, we get 1744 data-points, where a datapoint is of the format ([title, abstract, authors], research track label). We omitted collecting the introduction section (as proposed in the proposal) for two reasons: a) in real life setting, when submitting a paper to ACL, the title, abstracts and the author list have to be submitted a week earlier than submitting the full paper. And it is during this time that the research track of the paper is chosen too. So effectively, for predicting the research track, we can only use the title, abstract and the author list, and b) in BERT models, the maximum sequence length for accommodating the introduction section causes a barrier in training, as we didn't have enough compute resources to train BERT on long sequences (this will be discussed further in the approach section).

In the proposal, we said that we would be collecting data from ACL 2013 to 2021. We didn't collect data from ACL 2018 and before because we observed that the research track labels of these conferences were different than the more recent ones. For example, ACL 2013 has labels like 'Linguistic Creativity' and 'Transliteration/Alignment', and we don't see papers under these tracks anymore.

Table 2 shows two examples of our dataset. These examples depict the how tough the task of research track prediction is, and how clean we can expect the collected dataset to be. Both the examples create a new dataset, but one is labeled as 'Computational Social Science, Social Media, and Cultural Analysis', and the other is labeled as 'Resources and Evaluation'. In an ideal scenario, the first example should have had both the tags. Unclean data is another reason why this task can be challenging for neural networks.

We randomly shuffle the 1744 data points and divide them into training-validation-test splits of sizes 1386, 175 and 183 respectively. The same splits are used across all our experiments with our models (both baseline and RPT), to maintain uniformity. We maintain the same class distribution

Example 1	**Example 2**
Title: CONAN - COunter NArratives through Nichesourcing: a Multilingual Dataset of Responses to Fight Online Hate Speech	**Title**: ChID: A Large-scale Chinese IDiom Dataset for Cloze Test
Authors: Yi-Ling Chung, Elizaveta Kuzmenko, Serra Sinem Tekiroglu, Marco Guerini	**Authors**: Chujie Zheng, Minlie Huang, Aixin Sun
Abstract: Although there is an unprecedented effort to provide adequate responses in terms of laws and policies to hate content on social media platforms, dealing with hatred online is still a tough problem. Tackling hate speech in the standard way of content deletion or user suspension may be charged with censorship and overblocking. One alternate strategy, that has received little attention so far by the research community, is to actually oppose hate content with counter-narratives (i.e. informed textual responses). In this paper, we describe the creation of the first large-scale, multilingual, expert-based dataset of hate-speech/counter-narrative pairs. This dataset has been built with the effort of more than 100 operators from three different NGOs that applied their training and expertise to the task. Together with the collected data we also provide additional annotations about expert demographics, hate and response type, and data augmentation through translation and paraphrasing. Finally, we provide initial experiments to assess the quality of our data.	**Abstract**: Cloze-style reading comprehension in Chinese is still limited due to the lack of various corpora. In this paper we propose a large-scale Chinese cloze test dataset ChID, which studies the comprehension of idiom, a unique language phenomenon in Chinese. In this corpus, the idioms in a passage are replaced by blank symbols and the correct answer needs to be chosen from well-designed candidate idioms. We carefully study how the design of candidate idioms and the representation of idioms affect the performance of state-of-the-art models. Results show that the machine accuracy is substantially worse than that of human, indicating a large space for further research.
Label: Computational Social Science, Social Media and Cultural Analytics	**Label**: Resources and Evaluation

Table 2: Two examples from our dataset

ratio in the three splits.

5 Baselines

In order to test how effective the BERT representations are, we compare with a similar overall setup, where the language model that we use to get the vector representations of title, abstract, and authors is different. We compare RPT with with document representation techniques like tf-idf (Baseline$_{tf-idf}$), and LSTMs (Hochreiter and Schmidhuber, 1997) using pretrained Word2Vec (Mikolov et al., 2013) word vectors (Baseline$_{RNN}$). We chose these baseline models as they are the simplest models we can use to predict the label, as done by RPT.

In (Baseline$_{tf-idf}$), we represent the concatenated title and abstract string using the TF-IDF vector. We built the TF-IDF vectorizer on the training set, and get the representations of the documents in the training, validation and the test split using this vectorizer. These representations are then used to train a Multinomial Naive Bayes classifier. We use this baseline to see how the basic document representation techniques fair on the task of research track prediction. We chose TF-IDF in particular because it relies on terms and their inverse frequencies/importances, similar to topic modeling approaches to this problem.

In Baseline$_{RNN}$ approach, we use the concatenated title and abstract as the input. We then extract the pre-trained word embeddings from the Word2Vec model which was pre-trained on Google News dataset[2]. We decided to use pre-trained Word2Vec embeddings to have a fair com-

[2] https://code.google.com/archive/p/word2vec/

parison with the RPT models, which use the pre-trained BERT models. Also, since the word embeddings constitute most of the parameters of a recurrent neural network, and given that we don't have a lot of data, we use pre-trained embeddings to improve the generalization performance on unseen data. Once we have the Word2Vec vectors for all the tokens in the input, we feed them to a 100 unit LSTM network with a 20-unit fully connected softmax layer at the end, to output the probabilites of the 20 classes. We train this LSTM for 100 epochs with batch size of 32 and cross-entropy loss, and choose the model which gives the best accuracy on validation set.

In both the baseline models, we use the same train/validation/test splits as described in section 4.

6 Approach

Since the 20 research-track tags are distinct from each other, and each paper is tagged with just one track, we model this problem as a multi-class classification problem. We fine-tune a BERT base model (uncased) (Devlin et al., 2019) to predict the class label. This is a 12 transformer layer architecture, with a softmax unit at the end, to predict the individual class probabilities. The class with the highest output probability is taken as the predicted class tag for the input. In this paper, this model architecture is called as RPT (Research Paper Tagger). We now show the different variants of RPT that we experimented with.

We have three modalities of information in the input (title, author names, and abstract), and we wanted to see the contribution of each of the modality in predicting the class label. So we fine-tune a BERT model on all the three modalities combined (RPT$_{Title-Abstract-Authors}$), three BERT models on all the three pairings of modalities (RPT$_{Title-Abstract}$), RPT$_{Title-Authors}$, and RPT$_{Abstract-Authors}$, and three BERT models on the three individual modalities (RPT$_{Title}$, RPT$_{Abstract}$, and RPT$_{Authors}$).

Note that we have performed a thorough hyperparameter tuning and hopefully squeezed out the best performance over all the models. The hyperparameters that we tested are the number of epochs (for early stopping, over a range [5, 10, 15, 20, 25, 30]), batch size (over a range [4, 8, 16, 32]), and learning rate (over a range [10^{-5}, $3*10^{-5}$, $5*10^{-5}$, $7*10^{-5}$, 10^{-4}]). We perform a random search over 60 combinations over this hyper-parameter space, and choose those which give the best top-1 accuracy on the validation set. For almost all of the models, we notice that the default values of learning rate ($5*10^{-5}$) gave the best validation accuracy, and that the most of the models were saturating/over-fitting after 10-15 epochs of training (see figure 1).

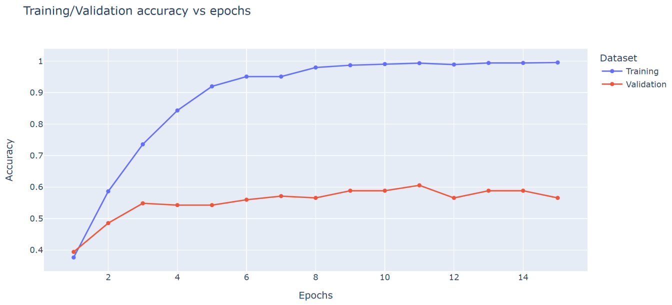

Figure 1: Accuracy vs epochs of the RPT$_{Title}$ model. This model saturates after 11 epochs. All the models generally saturate or start overfitting after 10-15 epochs of training.

6.1 Results and analysis

The classification metrics for the test set are shown in 3. We analyse our models using top-1, top-3, top-5 accuracies, and the weighted averages of precision, recall and F-1 scores, weighted according to the class distribution. We decide to produce the top-3 and top-5 accuracies as getting a high top-1 accuracy on a classification task with 20 classes is a tough task. We can use our model to output 3 or 5 track recommendations based on the title, abstract and author names, and top-3 and top-5 accuracies are indicative of how good the 3 or 5 recommendations would be.

Compared to the baselines, our models improve significantly on all the metrics. This was expected, as BERT models are higher capacity models with much larger number of parameters than LSTMs and tf-idf based classifiers. **For this particular task of research track prediction on our dataset, there are no existing baselines implemented by other works, so we coded up and trained the two baselines from the scratch.**

One thing to note from these results is that the author names provide very little information to predict the research track. RPT$_{Authors}$ is only 18.57% accurate, and the top-3 and top-5 accuracies are also low (33.33% and 42.07% respectively). This was anticipated by us; since

Model	Top-1 acc	Top-3 acc	Top-5 acc	Precision	Recall	F-1 score
Baseline$_{tf-idf}$	36.61%	49.18%	56.83%	0.24	0.36	0.25
Baseline$_{RNN}$	53.55%	74.86%	81.96%	0.52	0.53	0.51
RPT$_{Authors}$	18.57%	33.33%	42.07%	0.16	0.18	0.17
RPT$_{Title}$	61.20%	78.14%	87.43%	0.62	0.61	0.61
RPT$_{Abstract}$	68.30%	81.42%	85.24%	0.70	0.68	0.66
RPT$_{Title-Authors}$	61.20%	76.50%	86.33%	0.61	0.61	0.60
RPT$_{Authors-Abstract}$	63.38%	83.60%	87.97%	0.64	0.63	0.61
RPT$_{Title-Abstract}$	69.39%	79.23%	84.15%	0.71	0.64	0.64
RPT$_{Title-Authors-Abstract}$	69.94%	79.23%	85.79%	0.71	0.69	0.68

Table 3: Classification metrics of different variations of RPT, and of the baseline models.

it is very tough for a BERT model to store this world knowledge, unless the author is really well known in a particular field of work. We also notice that Author names act as noise sometimes, when combined with other modalities; when combined with abstract, we see a sharp drop in accuracy from 68.30% of RPT$_{Abstract}$ to 63.38% of RPT$_{Authors-Abstract}$. We even see a drop in the precision, recall and F-1 score in these two models. Similar trend is observed in RPT$_{Title}$ and RPT$_{Title-Authors}$; we see a drop in precision, recall, F-1 score. However, we notice a different trend when we move from RPT$_{Title-Abstract}$ to RPT$_{Title-Authors-Abstract}$; the top-1 accuracy increases slightly from 69.39% to 69.94%, and so do the precision, recall and F-1 scores. We couldn't comprehend why the trend is opposite in this case.

Title and abstract provide complimentary information to help predict research track. We infer this from the improvement in classification metrics when we move from RPT$_{Title}$ to RPT$_{Title-Abstract}$ (top-1 accuracy improves sharply from 61.20% to 69.39%, precision improves from 0.62 to 0.71, recall improves from 0.61 to 0.64, and F-1 score improves from 0.61 to 0.64). We also notice that although title and abstract provide complimentary information, most of the contribution comes from the abstract, as we don't see much improvement in classification metrics when we compare RPT$_{Abstract}$ and RPT$_{Title-Abstract}$.

When combining modalities, we didn't separate them using the [SEP] token, we separated them using full stop. This is because the BERT model has not seen more than one [SEP] tokens during the pre-training, and when we combine title, abstract and the authors, we need two [SEP] tokens as the delimiters. We verified this empirically, the accuracy on the validation set when using [SEP] was always lower than when using full stop as the delimiter.

6.2 What's different compared to the proposed technique?

In the proposal, we said that we would be training a slightly different architecture for RPT$_{Title-Authors-Abstract}$, where, for each of the three models RPT$_{Title}$, RPT$_{Abstract}$, and RPT$_{Authors}$, we take the CLS representation from the last layer of BERT, compose them (by concatenating) into a single vector, and pass this vector to a deep classifier to output the research track predictions. The proposed architecture is shown in figure 2. We proposed this architecture because it doesn't require excessive memory to train, as is required by RPT$_{Title-Authors-Abstract}$. The main memory consumption occurs because of the longer sequence length, when we train the RPT$_{Title-Authors-Abstract}$ on the concatenated [Title, Abstract, Authors] sequence. Where as in the originally proposed models, we train the individual RPTs (RPT$_{Title}$, RPT$_{Authors}$, and RPT$_{Abstract}$) separately, thereby requiring a lesser sequence length for each of the individual RPTs (hence lesser memory). Although the originally proposed model requires lesser memory to train, we run into a significant learning issue, and we didn't anticipate these issues during the proposal. The CLS vector for the BERT base model is of length 768. When we concatenate the three CLS vectors, it gives us a vector of length 2304. This 2304 length vector is used as an input to the deep classifier. Because this vector is very long, curse of dimensionality plays its hand. We only have 1386 training points, which is less than the vector length 2304. When trained, the deep classi-

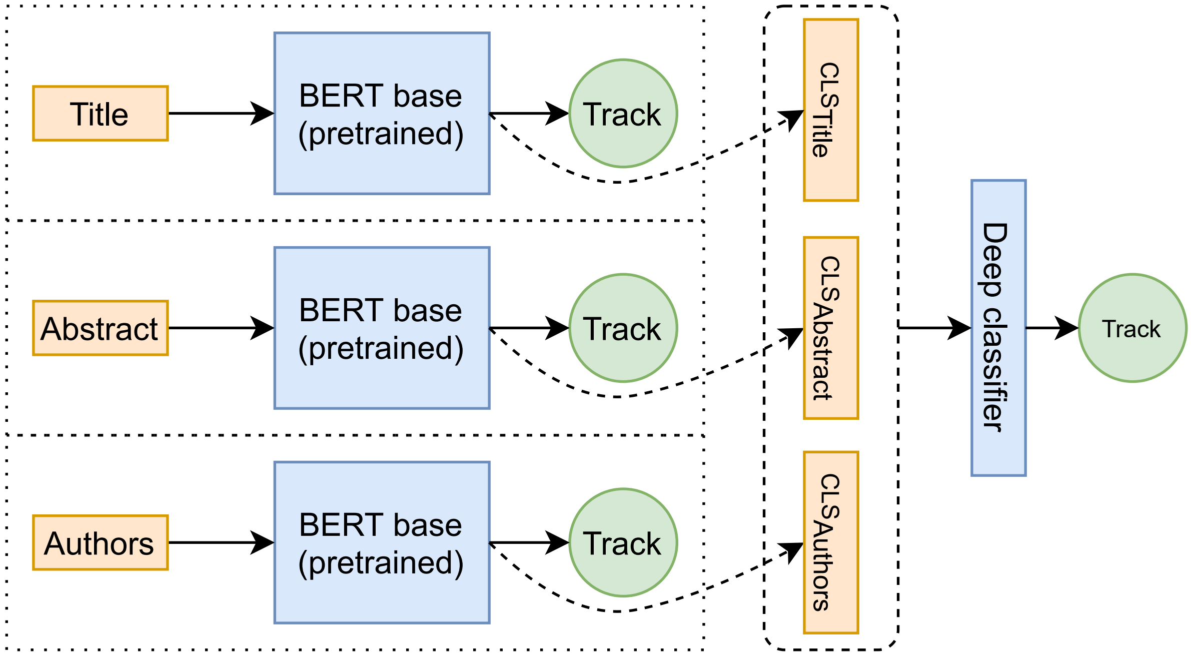

Figure 2: The originally proposed RPT model ($\text{RPT}_{Title-Authors-Abstract}$). The three BERT base models are trained independently, and the CLS token from each of them is extracted. The three CLS tokens are concatenated, and passed to a deep classifier, that we train from scratch, to predict the research track label.

fier quickly starts to overfit on the training data, and doesn't show a validation accuracy of more than 50%. To circumvent this problem, we tried other composition techniques like addition and weighted addition (so that the composed vector is of a shorter length 768), but this too didn't improve the accuracy. This is why we decide to discard this approach, and use the approaches described before.

6.3 RPT works correctly on the RPT paper

We predicted the research track of our own paper, using our own model $\text{RPT}_{Title-Authors-Abstract}$. As a reminder, in the abstract, we proposed to introduce a dataset containing the papers of ACL 2019, 2020 and 2021, and we proposed to build RPT, which would be trained on this dataset, to predict the research tags, given the title, abstract and the author names. $\text{RPT}_{Title-Authors-Abstract}$ predicts the top-3 labels for our paper as 'NLP-application', 'Resources and Evaluation', and 'Generation'. Our paper is indeed an application of NLP; it uses a BERT based model to predict the research tags, and it is aptly tagged as 'Resources and Evaluation', as we propose to release a dataset of ACL papers and their research tracks. RPT works correctly on the RPT paper, how cool is that?

6.4 Challenges faced

The major challenge we faced was a dire shortage of compute resources. We train the models on Google Colab (Bisong, 2019), and on our personal laptops. One of us has a 6 GB NVIDIA RTX 2060 GPU, and that proved helpful in faster training of the RPT models. Nevertheless, the memory in both Google Colab, and in the personal laptop GPUs was less than sufficient to train the RPT models using the default batch size. Infact, for $\text{RPT}_{Title-Authors-Abstract}$, we had to set the batch size to 2, to train the model. Lesser batch sizes usually don't give good generalization over unseen data, and we noticed this in $\text{RPT}_{Title-Authors-Abstract}$ when it was trained on a batch size of 2 (the validation accuracy was as low as 58%). To circumvent this issue, we used the technique of gradient accumulation, where we accumulate the parameter gradients for multiple steps (or multiple batches), and update the model only in a few steps. For example, with the limitation of batch size 2, if we want to realize training with batch size 16, we accumulate the model gradients over every step and update the model parameter after every 8 steps. This technique gives us an effective batch size of 16 despite having limited memory, at the cost of longer training time.

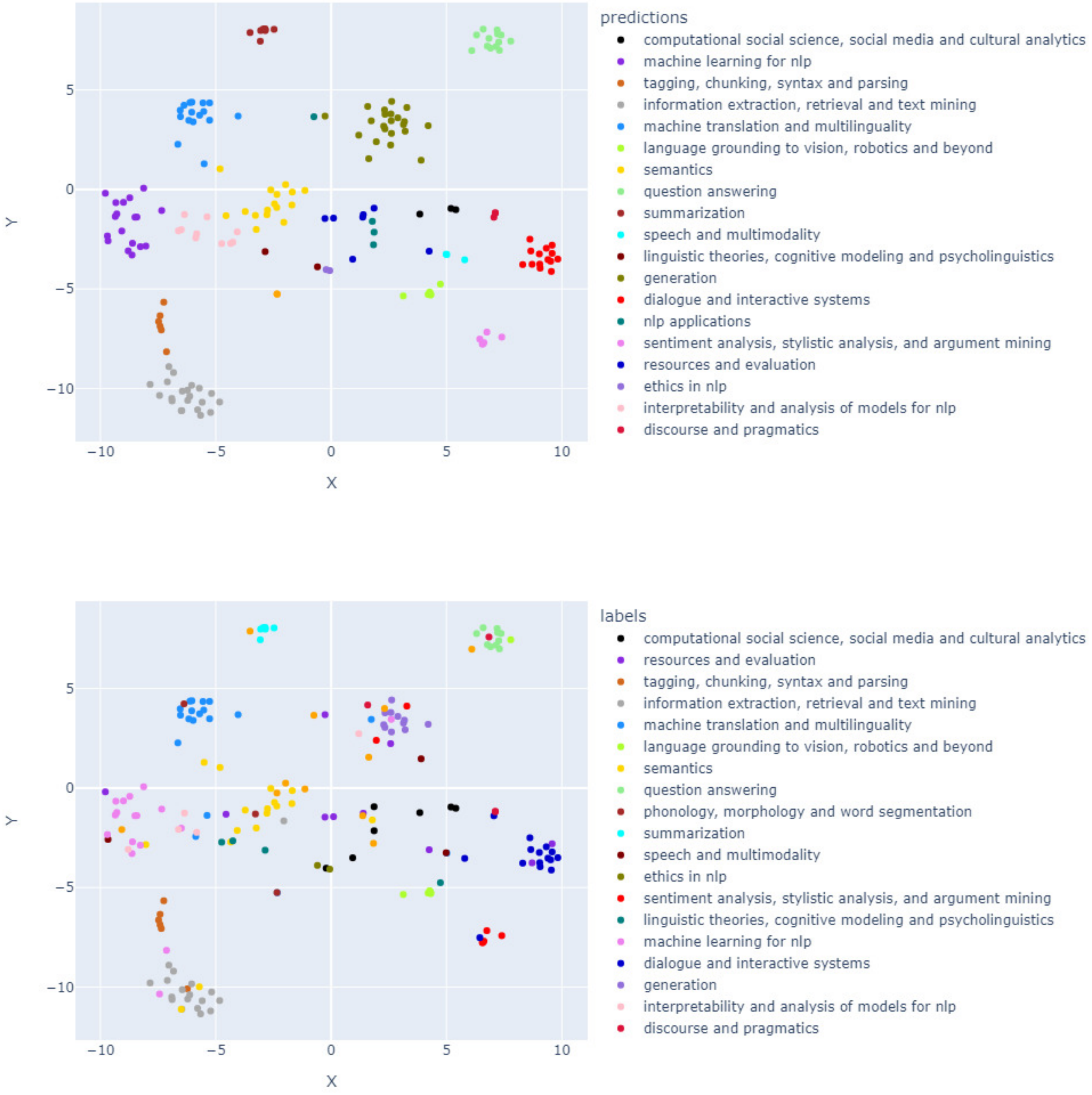

Figure 3: 2D T-SNE representation of the [CLS] vector obtained by the RPT model.

6.5 Coding and implementation details

For data extraction, we didn't use any specialized libraries or tools, the default Python utilites were sufficient for this. We used Hugging Face's (Wolf et al., 2020) pretrained BERT base model (uncased) for sequence classification. The training loop is written in PyTorch (Paszke et al., 2019), and the data preparation for the same uses NumPy (Harris et al., 2020) as Pandas (pandas development team, 2020). To plot the training metrics, we used the plotting library Plotly (Inc., 2015). For RNNs, we used Tensorflow Keras's (Abadi et al., 2015) LSTM module. And for TF-IDF, we used the built in functions of scikit-learn (Pedregosa et al., 2011). As mentioned earlier, we trained our models on Google Colab (Bisong, 2019) and on personal laptop GPUs (6 GB RTX 2060).

RNN models and training was coded from scratch, and so was tf-idf's. We used an existing

Misclassified example 1	**Misclassified example 2**
Title: Which Linguist Invented the Lightbulb? Presupposition Verification for Question-Answering	**Title**: DialogueCRN: Contextual Reasoning Networks for Emotion Recognition in Conversations
Authors: Najoung Kim, Ellie Pavlick, Burcu Karagol Ayan, Deepak Ramachandran	**Authors**: Dou Hu, Lingwei Wei, Xiaoyong Huai
Abstract: Many Question-Answering (QA) datasets contain unanswerable questions, but their treatment in QA systems remains primitive. Our analysis of the Natural Questions (Kwiatkowski et al. 2019) dataset reveals that a substantial portion of unanswerable questions (approx 21%) can be explained based on the presence of unverifiable presuppositions. We discuss the shortcomings of current models in handling such questions, and describe how an improved system could handle them. Through a user preference study, we demonstrate that the oracle behavior of our proposed system that provides responses based on presupposition failure is preferred over the oracle behavior of existing QA systems. Then we discuss how our proposed system could be implemented, presenting a novel framework that breaks down the problem into three steps: presupposition generation, presupposition verification and explanation generation. We report our progress in tackling each subproblem, and present a preliminary approach to integrating these steps into an existing QA system. We find that adding presuppositions and their verifiability to an existing model yields modest gains in downstream performance and unanswerability detection. The biggest bottleneck is the verification component, which needs to be substantially improved for the integrated system to approach ideal behavior – even transfer from the best entailment models currently falls short.	**Abstract**: Emotion Recognition in Conversations (ERC) has gained increasing attention for developing empathetic machines. Recently, many approaches have been devoted to perceiving conversational context by deep learning models. However, these approaches are insufficient in understanding the context due to lacking the ability to extract and integrate emotional clues. In this work, we propose novel Contextual Reasoning Networks (DialogueCRN) to fully understand the conversational context from a cognitive perspective. Inspired by the Cognitive Theory of Emotion, we design multi-turn reasoning modules to extract and integrate emotional clues. The reasoning module iteratively performs an intuitive retrieving process and a conscious reasoning process, which imitates human unique cognitive thinking. Extensive experiments on three public benchmark datasets demonstrate the effectiveness and superiority of the proposed model.
Predicted label: Question Answering	**Predicted label**: Sentiment Analysis, Stylistic Analysis, and Argument Mining
Ground truth label: Discourse and Pragmatics	**Ground truth label**: Dialogue and Interactive systems

Table 4: Two examples from the test set which were misclassified by RPT$_{Title-Authors-Abstract}$

implementation to train RPT, this code was taken from the homework 1 of CS 685 Fall 21 course [3]. Of course, we made our own custom changes to the existing implementations to fit our needs.

On the GitHub repo https://github.com/arbaazQureshi/research_paper_tagger, we have put the code for data extraction and model training for all of our models. Since the saved models were too large to be imported on GitHub, we didn't put them there. We have kept our datasets too on GitHub, they can be found in the Data directory.

7 Error analysis

We analyse RPT$_{Title-Authors-Abstract}$ on the test set, as this model was giving the best performance

[3] https://colab.research.google.com/drive/1cTDpMUuJbxxaGEcNe11KQXLLweu8OA8r?usp=sharing

Title: Unsupervised Extractive Summarization-Based Representations for Accurate and Explainable Collaborative Filtering
Authors: Reinald Adrian Pugoy, Hung-Yu Kao
Abstract: We pioneer the first extractive summarization-based collaborative filtering model called ESCOFILT. Our proposed model specifically produces extractive summaries for each item and user. Unlike other types of explanations, summary-level explanations closely resemble real-life explanations. The strength of ESCOFILT lies in the fact that it unifies representation and explanation. In other words, extractive summaries both represent and explain the items and users. Our model uniquely integrates BERT, K-Means embedding clustering, and multilayer perceptron to learn sentence embeddings, representation-explanations, and user-item interactions, respectively. We argue that our approach enhances both rating prediction accuracy and user/item explainability. Our experiments illustrate that ESCOFILT's prediction accuracy is better than the other state-of-the-art recommender models. Furthermore, we propose a comprehensive set of criteria that assesses the real-life explainability of explanations. Our explainability study demonstrates the superiority of and preference for summary-level explanations over other explanation types.
Predicted label: Summarization
Ground truth label: NLP applications

Table 5: One example from the test set where RPT$_{Title-Authors-Abstract}$ predicted a more apt research track

on the validation set. We manually try to find patterns and commonalities among the data points which were incorrectly classified.

To visualise how well separated the [CLS] vectors are for each of the test set prediction that RPT does, we plot a scatter plot of the 2-D T-SNE of the [CLS] vector. The T-SNE (Van der Maaten and Hinton, 2008) algorithm is a dimensionality reduction algorithm. We get the 2-D T-SNE vectors for all of the 183 [CLS] vectors of the test set. Then, we plot a scatter plot of these T-SNE points, see the figure 3. To show how well the RPT$_{Title-Authors-Abstract}$ is clustering the test set inputs, we color the T-SNE points according to their predictions (see the top plot in figure 3). In order to see how representative the clusters are of the actual research track class labels, we colour them according to their actual labels (see the bottom plot in the figure 3). It is evident that RPT$_{Title-Authors-Abstract}$ has well-clustered [CLS] vectors for its predictions, but these [CLS] clusters are not always representative of the actual classes. For example, see the light green coloured cluster located at the top right of the bottom scatter plot. This light green cluster corresponds to the class 'Question Answering'. But there is a red point, whose actual label is 'Discourse and Pragmatics', but is wrongly classified as 'Question Answering' (we call this example as misclassified example 1). Similarly, see the red cluster at the bottom right of the bottom plot. This cluster corresponds to 'Sentiment Analysis, Stylistic Analysis, and Argument Mining'. But there's a dark blue point in this cluster, and it belongs to the class 'Dialogue and Interactive systems' (we call this example as misclassified example 2). We find commonalities in many misclassified examples. We talk about the two misclassified examples mentioned above in the following discussion.

Table 4 shows two example from the test set which were misclassified by RPT$_{Title-Authors-Abstract}$, as shown before. It is evident why these two examples were wrongly classified. The title and abstract of the first misclassified example contains many occurences of the phrase 'Question-Answering'. Also, the authors of this work propose an improved QA system which handles the unanswerable questions, by introducing presupposition; in short, they propose an improved QA model. This paper could also be tagged with 'Discourse and Pragmatics', but 'Question Answering' would be the most apt tag. As a small experiment, the group member that was analysing this misclassified example asked the other group members to predict the label for this data point. All three of the group members predicted the label as 'Question Answering'. Although this isn't a completely scientific way to prove this hypothesis, it certainly hints that even for humans, the label of this example would've been tough to predict correctly.

Now see the second misclassified example.

Both the title and abstract have the words 'emotion recognition', which is more closely related to the tag 'Sentiment Analysis, Stylistic Analysis, and Argument Mining' than 'Dialogue and Interactive systems'. Also, it isn't very clear by the abstract that the authors have worked on a dialogue and interactive system, there is no mention in the abstract or the title using which we can clearly say that it is a work on dialogue and interactive systems. It has to be noted however that 'Dialogue and Interactive Systems' was in the top-3 predictions of the $\text{RPT}_{Title-Authors-Abstract}$, for this example.

We manually peruse through all the 55 misclassified examples. We found that many misclassified examples were similar to the ones showed in table 4, where it is indeed tough, even for humans, to predict correctly. There is one specific set of 9 examples whose actual labels are 'NLP applications', but $\text{RPT}_{Title-Authors-Abstract}$ was predicting a more specific and apt track. See table 5 for the title, author names and abstract of the paper. This paper clearly talks about producing extractive summaries. We believe that this paper was aptly classified by $\text{RPT}_{Title-Authors-Abstract}$ as 'Summarization'. Out of the 9 examples where the label is 'NLP applications', we believe that the model was predicting a better and a more apt label in 5 examples.

8 Contributions of group members

- **Syed Arbaaz Qureshi**: Data collection, Dataset creation, RPT models training and evaluation, error analysis, paper writing.

- **Avani Jindal**: Data collection, Baseline_{tfidf} training and evaluation, Baseline_{RNN} training and evaluation, error analysis, paper writing

- **Muskan Jain**: Data collection, Baseline_{tfidf} training and evaluation, error analysis, paper writing.

- **Sahil Jindal**: Data collection, Baseline_{RNN} training and evaluation, Error analysis, paper writing.

9 Conclusion

Working on this project was indeed a great journey and we gained practical experience with NLP tools and techniques. One of the major takeaways from this project was that using a pre-trained weights like those of BERT or Word2Vec improves the performance by a great extent, as observed by comparing baseline models with RPT. We noticed a huge difference when we trained Baseline_{RNN} on freshly initialized word embeddings versus when we used pre-trained Word2Vec embeddings. When using untrained word embeddings, we achieved a maximum validation accuracy of 38.86%. To improve this, we used the pre-trained model Word2Vec to get the embeddings, and the validation accuracy shot up to 52.57%. Another thing was gradient accumulation where we were able to train RPT with a bigger batch size. We weren't able to train the $\text{RPT}_{Title-Authors-Abstract}$ model on a batch size greater than 2. But with gradient accumulation, we were able to accumulate the gradients of 8 steps and then update them in the 8th step. Even though this is done at the cost of training time (because only 2 examples are processed at a time), we get a higher effective batch size. Lastly, it is always a good idea to reuse a standard implementation, instead of coding everything up from scratch. This reduces the chance of bugs in the code, thereby facilitating fast prototyping.

The biggest difficulty we faced was a shortage of compute resources. We needed more memory to train RPT effectively, on larger batch sizes. As stated earlier, we circumvented this problem by using the technique of gradient accumulation.

We discussed in results and analysis (section 6.1), that unless the author is a really well known person in a field of work, it is difficult for BERT to model the world knowledge of associating the author with a research track. One way to get more out of author names is by providing more information about the author to the RPT. We can extract more data about the past work of the authors (like his previous publications, his co-authors), and use it along with our existing data to make better predictions. The previous publications of an author gives insight into what sub-domain of a broad research area the author is assoicated with, thereby making the prediction of research track more easy.

References

Abadi, M., Agarwal, A., Barham, P., Brevdo, E., Chen, Z., Citro, C., Corrado, G. S., Davis, A., Dean, J., Devin, M., Ghemawat, S., Goodfellow, I., Harp, A., Irving, G., Isard, M., Jia, Y., Jozefowicz, R., Kaiser, L., Kudlur, M., Levenberg, J., Mané, D., Monga, R., Moore, S., Murray, D., Olah, C., Schuster, M., Shlens, J., Steiner, B., Sutskever, I., Talwar, K., Tucker, P., Vanhoucke, V., Vasudevan, V., Viégas, F., Vinyals, O., Warden, P., Wattenberg, M., Wicke, M., Yu, Y., and Zheng, X. (2015). TensorFlow: Large-scale machine learning on heterogeneous systems. Software available from tensorflow.org.

Abuhay, T. M., Kovalchuk, S. V., Bochenina, K., Mbogo, G.-K., Visheratin, A. A., Kampis, G., Krzhizhanovskaya, V. V., and Lees, M. H. (2018). Analysis of publication activity of computational science society in 2001–2017 using topic modelling and graph theory. *Journal of Computational Science*, 26:193–204.

A.K., M., K., N., and J., R. (2000). Automating the construction of internet portals with machine learning. *Information Retrieval*, 3:127–163.

Bafna, P., Pramod, D., and Vaidya, A. (2016). Document clustering: Tf-idf approach. pages 61–66.

Bisong, E. (2019). *Google Colaboratory*, pages 59–64. Apress, Berkeley, CA.

Blei, D. M., Ng, A. Y., and Jordan, M. I. (2003). Latent dirichlet allocation. *J. Mach. Learn. Res.*, 3(null):993–1022.

Cortes, C. and Vapnik, V. (1995). Support vector networks. *Machine Learning*, 20:273–297.

Devlin, J., Chang, M.-W., Lee, K., and Toutanova, K. (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.

Harris, C. R., Millman, K. J., van der Walt, S. J., Gommers, R., Virtanen, P., Cournapeau, D., Wieser, E., Taylor, J., Berg, S., Smith, N. J., Kern, R., Picus, M., Hoyer, S., van Kerkwijk, M. H., Brett, M., Haldane, A., del Río, J. F., Wiebe, M., Peterson, P., Gérard-Marchant, P., Sheppard, K., Reddy, T., Weckesser, W., Abbasi, H., Gohlke, C., and Oliphant, T. E. (2020). Array programming with NumPy. *Nature*, 585(7825):357–362.

Hochreiter, S. and Schmidhuber, J. (1997). Long short-term memory. *Neural Comput.*, 9(8):1735–1780.

Inc., P. T. (2015). Collaborative data science.

Kandimalla, B., Rohatgi, S., Wu, J., and Giles, C. L. (2021). Large scale subject category classification of scholarly papers with deep attentive neural networks. *Frontiers in Research Metrics and Analytics*, 5:31.

Korlapati, M., Ravipati, T., Jha, A. K., and Prakash, K. B. (2019). Categorizing research papers by topics using latent dirichlet allocation model. *INTERNATIONAL JOURNAL OF SCIENTIFIC & TECHNOLOGY*, 8:1442–1446.

Lenc, L. and Král, P. (2018). Deep neural networks for czech multi-label document classification. In Gelbukh, A., editor, *Computational Linguistics and Intelligent Text Processing*, pages 460–471, Cham. Springer International Publishing.

Liu, S., Tao, H., and Feng, S. (2019). Text classification research based on bert model and bayesian network. In *2019 Chinese Automation Congress (CAC)*, pages 5842–5846.

Mcauliffe, J. and Blei, D. (2008). Supervised topic models. In Platt, J., Koller, D., Singer, Y., and Roweis, S., editors, *Advances in Neural Information Processing Systems*, volume 20. Curran Associates, Inc.

Mikolov, T., Chen, K., Corrado, G., and Dean, J. (2013). Efficient estimation of word representations in vector space. *arXiv preprint arXiv:1301.3781*.

Mosbach, M., Andriushchenko, M., and Klakow, D. (2021). On the stability of fine-tuning {bert}: Misconceptions, explanations, and strong baselines. In *International Conference on Learning Representations*.

pandas development team, T. (2020). pandas-dev/pandas: Pandas.

Paszke, A., Gross, S., Massa, F., Lerer, A., Bradbury, J., Chanan, G., Killeen, T., Lin, Z., Gimelshein, N., Antiga, L., Desmaison, A., Kopf, A., Yang, E., DeVito, Z., Raison, M., Tejani, A., Chilamkurthy, S., Steiner, B., Fang, L., Bai, J., and Chintala, S. (2019). Pytorch: An imperative style, high-performance deep learning library. In Wallach, H., Larochelle, H., Beygelzimer, A., d'Alché-Buc, F., Fox, E., and Garnett, R., editors, *Advances in Neural Information Processing Systems 32*, pages 8024–8035. Curran Associates, Inc.

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

Sammut, C. and Webb, G. I., editors (2010). *TF–IDF*, pages 986–987. Springer US, Boston, MA.

SW. Kim, J. G. (2019). Research paper classification systems based on tf-idf and lda schemes. *Human-centric Computing and Information Sciences*, 30.

Trstenjak, B., Mikac, S., and Donko, D. (2014). Knn with tf-idf based framework for text categorization. *Procedia Engineering*, 69:1356–1364. 24th DAAAM International Symposium on Intelligent Manufacturing and Automation, 2013.

Van der Maaten, L. and Hinton, G. (2008). Visualizing data using t-sne. *Journal of machine learning research*, 9(11).

Vaswani, A., Shazeer, N., Parmar, N., Uszkoreit, J., Jones, L., Gomez, A. N., Kaiser, L., and Polosukhin, I. (2017). Attention is all you need. *CoRR*, abs/1706.03762.

Wolf, T., Debut, L., Sanh, V., Chaumond, J., Delangue, C., Moi, A., Cistac, P., Rault, T., Louf, R., Funtowicz, M., Davison, J., Shleifer, S., von Platen, P., Ma, C., Jernite, Y., Plu, J., Xu, C., Scao, T. L., Gugger, S., Drame, M., Lhoest, Q., and Rush, A. M. (2020). Transformers: State-of-the-art natural language processing. In *Proceedings of the 2020 Conference on Empirical Methods in Natural Language Processing: System Demonstrations*, pages 38–45, Online. Association for Computational Linguistics.

Yu, Q., Wang, Z., and Jiang, K. (2021). Research on text classification based on BERT-BiGRU model. *Journal of Physics: Conference Series*, 1746(1):012019.