Joint Modeling of Humor and Offense

Gabriel Brookman
gbrookman@umass.edu
Nicholas Samoray
nsamoray@umass.edu
Akshay Gugnani
agugnani@umass.edu
Brian Zylich
bzylich@umass.edu

1 Problem statement

With the advancement in deep learning methods, NLP tasks like sentiment analysis and opinion mining have achieved high accuracy, however some salient forms of figurative language such as humor are relatively less explored.

Being able to infer humor and offense with a high accuracy can help improve and lead to better performance on downstream applications, such as content moderation, sentiment analysis, etc. This would have great underlying applications in various industries, for instance to better understand product tweets, reviews and feedback. However, the task of humor detection is not simple.

What makes identifying humor hard? Humor can consist of styles ranging from sarcastic to slapstick comedy, and it could factor in both individual choices and underlying cultures. Context, sounds and vision or any combination of these can be key in building to a punchline, or in cleverly steering the joke to a less-expected punchline (Cai, 2019). Humor appreciation is also a highly subjective phenomenon, with age, gender and socioeconomic status known to have an impact on the perception of a joke.

Humor, unlike other classification tasks can be subject to various factors. For instance, in the classification of an image, which may or may not contain a cat, the classification is binary. However, as explained by Meaney (2020), when we look at humor we need to also factor subjective factors like demographic details. The 3 broad distinctions Meaney makes are
(1) Humor can differ between cultures,
(2) Humor can also differ within cultures, and
(3) Humor differs within the same person.

In our work, we will try to address humor in these subjective contexts. Further, we will be participating in the SemEval'21 Task 7: HaHackathon (J. A. Meaney, 2020), which challenges participants to develop a model for humor detection which incorporates the subjectivity associated with humor across different demographic groups, and to provide classifications and ratings based on age and gender. The organizers updated the event after our initial project proposal and we discuss the changes in the following section that expands on our goals and tasks.

Our code base is on our Github [1] repository, however it is a private repository as the SemEval event is ongoing till Jan'21. We have however attached the zipped repository with our submission on Gradescope.

2 What you proposed vs. what you accomplished

We had originally planned to participate in the HaHackathon task, which we described in our project proposal. However, the organizers decided to update the task due to limited demographic data they could capture. We decided to follow the new tasks. The primary difference is that the new tasks do not contain or leverage specific demographic data. We explain both the original and updated tasks below.

2.1 HaHackathon Tasks - Original Proposal

The HaHackathon competition was originally divided into two main tasks. As in prior humor prediction tasks, the first task involved classifying and scoring texts according to their humor/offensiveness based on averaged ratings from multiple raters. Then, in recognition of the subjectivity of humor and offensiveness based on demographic factors, the second task focused on predicting how humorous/offensive a text is for people from different age groups and genders.

[1] https://github.com/bzylich/humor-by-demographic

The subtasks were defined as follows:
Task 1a: Given a text, predict if the class is humorous and/or offensive, or not.
Task 1b: If the text is classified as humorous and/or offensive, predict how humorous and/or offensive it is.
Task 2a: Given a text, predict if the class is offensive or not, for each age group and gender.
Task 2b: If the text is classed as humorous and/or offensive, predict how humorous and/or offensive it is for each age group and gender.

2.2 HaHackathon Tasks - Updated Proposal

The tasks of the competition were changed after we submitted the initial proposal. The updated sub-tasks are the following:
Task 1a: Given a text, predict if the class is humorous and/or offensive, or not.
Task 1b: If the text is classified as humorous, predict how humorous it is (scale of 0-5).
Task 1c: Given a humorous text, predict if the humor is controversial or not.
Task 2: Predict how generally offensive the text is (scale of 0-5), whether it is humorous or not.

Thus, instead of breaking humor and offensiveness down by demographics, they introduce this humor controversy label which attempts to capture the idea of differences in perception of humor but at a very general level that prevents us from using some of the methods we initially proposed. Nevertheless, our work explores and attempts to develop the best model for the SemEval event described above.

We proposed to do the tasks mentioned in Section 2.2. We were able to complete and develop models for all the 4 tasks. We are able to achieve submissions with significant performance in all of them. Some of our approaches, such as intermediate finetuning, did not perform as well on the tasks as we had hoped for- we discuss our learning's related to this in more details in Section7 and how we overcame the challenges. As of the end of day of the submission of the project report, our team holds a position in the top 3 for each of the tasks[2], and we hold the first rank for 3 of the 4 tasks, among 21 other competing teams. We discuss this in more detail in the report.

The rest of the report is divided into related work (Section 3), descriptions of our datasets (Section 4), baselines (Section 5), our approach (Section 6), error analyses (Section 7), group member contributions (Section 8), and conclusion (Section 9).

3 Related work

The challenge of humor detection has gained some traction since 2017. Meaney (2020) explains in their proposal that prior work has explored humor detection as an objective task, where they average all annotations given to a joke, to give a single classification or rating. This treats humor as an objective concept, rather than the subjective phenomenon that it is in reality.

The authors of (Badlani et al., 2019) explain how text in reviews requires a deep semantic understanding since a lot of such text can be sarcastic, humorous, or hateful. A vanilla sentiment analysis would fail to perform well in such cases. They propose a two-step model that extracts features pertaining to sarcasm, humor, hate speech, and sentiment in the first step, and then feeds them in conjunction to inform sentiment classification in the second step. Their work is quite sensitive to catching negative sentiment, however, it does not do as well when sentiment changes halfway through the text. It also does not leverage the subjectivity in terms of demographic factors in their ensemble model.

Yang et al. (2015) explore computational models to discover the structures behind humor, recognize humor and even extract humor anchors. Their work explores and identifies several semantic structures behind humor and designs sets of features for each structure. They then propose a Maximal Decrement method to automatically extract anchors that enable humor in a sentence.

Earlier work, like that of Donahue et al. (2017), propose systems which utilize recurrent deep learning methods with dense embeddings to predict humorous tweets. In order to factor both meaning and sound in their analysis, they use GloVe embeddings which are combined with a novel phonetic representation to serve as input to an LSTM component. They use a character-based CNN model, and an XGBoost to achieve an accuracy of 0.675 on their data.

Hossain et al. (2020) hosted the SemEval'20 event for humor detection in news headlines. The event challenged participants to classify whether

[2]https://competitions.codalab.org/competitions/27446#results

an original headline or an altered headline is funnier and rate the funniness of the edited headline on a 0-3 humor scale. The winning teams (Morishita et al., 2020) combined the predictions of several hyperparameter tuned versions of models using regression as sentence pair regression and exploited an ensemble method of the pre-trained language models BERT , GPT-2, (Radford et al., 2019) RoBERTa, (Liu et al., 2019) , XLNet (Yang et al., 2019) , Transformer-XL and XLM (Dai et al., 2019) to arrive at the final prediction.

The author Annamoradnejad (2020) demonstrates an accuracy of 98.1%. In their work they use BERT (Devlin et al., 2018) to generate tokens and sentence embedding for texts, and then send embedding outputs as input to a two-layered neural network that predicts the target value. They also discuss that the combination of RNN-CNN was not successful in the task compared to their selected CNN model. Mao and Liu (2019) is another work that uses BERT on crowd-annotated tweets, classify if a tweet is a joke or not and predict a funniness score value for a tweet. The work of Weller and Seppi (2019) explores extending humor detection capability by trying to assess whether or not a joke is humorous. They attempt to do this by building a model that learns to identify humorous jokes based on ratings gleaned from Reddit pages. They use a transformer architecture in order to learn sentence context. They demonstrate the effectiveness of their approach and show results that are comparable to human performance.

Similar to humor detection, there has been some work to explore offense in text. SemEval '19 had a task (Zampieri et al., 2019), aimed at identifying and categorizing offensive language in social media. The top performing teams used ensembles of OpenAI GPT, Random Forest, the Transformer, Universal encoder, ELMo, and combined embeddings from fastText and their own custom embeddings. Another top team also used linear models, LSTM, and pre-trained BERT with finetuning on the dataset.

Most of the related work does not account for the subjectivity of humor nor do they leverage demographic information like age, gender, or socioeconomic factors. To our knowledge, our work would be one of the first to explore humor detection along these lines of subjectivity, motivated by the SemEval'21 task (J. A. Meaney, 2020).

4 Your dataset

4.1 Data

We use three types of datasets in this work: the HaHackathon competition dataset, datasets for offensive text detection, and datasets for humor detection. We describe each of these in the following subsections.

4.1.1 HaHackathon Competition Dataset

In our work we use the dataset released by the SemEval event[3]. The training dataset consists of 8000 texts. For our work we have created a randomized 90-10 train:development split of 7200 training examples and 800 sentences for model development. We discuss more details of this dataset in the annotations section. In addition, the competition has its own development dataset of 1000 texts. This dataset does not contain labels and evaluation is only available by submitting predictions to the leaderboard. We use this as our final test set since the actual test set for the competition is not released until after the project deadline.

4.1.2 Humor Datasets

In order to tackle pretraining for the humor component of the competition, we rely on two datasets. 200k Short Texts for Humor Detection (Annamoradnejad, 2020), and a self-compiled dataset of jokes and other texts scraped from Reddit.

The 200k Short Texts for Humor Detection dataset consists of a collection of 200,000 short text snippets, which were then each labeled as either humorous or not humorous, with an even split between the two classes. The non-humorous texts were scraped from news headlines from the Huffington Post while the humorous texts were taken from Reddit communities such as /r/jokes and /r/cleanjokes.

The other dataset was one which we compiled ourselves, consisting of 200,000 snippets of text scraped from various reddit communities. This was done primarily to address shortcomings we noticed in the 200k Short Texts dataset; namely the limited range of lengths for jokes and the singular source for the negative examples. For the positive examples of humor, we scraped the /r/jokes subreddit, and kept information such as the submissions total score, as well as its "upvote ratio". This ratio represents the proportion

[3]https://competitions.codalab.org/competitions/26083

of positive votes to negative votes, with a lower ratio indicating that more users found a submission to be unfunny, spam, or irrelevant. For negative examples, we scraped various subreddits, focusing on /r/reddit.com and /r/worldnews. While now defunct, the reddit.com subreddit offered a general catch all for submissions for reddit that wouldn't fit elsewhere. This included news headlines, generic conversation topics, and meta-commentary. Since the /r/jokes community existed at the same time, the headlines on /r/reddit.com were unlikely to be structured as a joke, and thus were used as negative examples of humor. This was combined with /r/worldnews text which focused on news headlines, but from a variety of sources rather than the singular source the 200k Short Texts dataset used.

4.1.3 Offense Datasets

The Offensive Language Identification Dataset (OLID) (Zampieri et al., 2019) is one dataset for identifying offensive language in texts, specifically tweets. It contains 14,200 tweets as well as annotations about whether or not (binary classification) they are offensive, which was used for pretraining for the offensiveness rating regression task.

Another dataset we used for the offensive text rating task was the Hate Speech and Offensive Language dataset (Davidson et al., 2017). This dataset consists of 26,953 tweets as well as labels corresponding to how many crowdflower users labeled them as hateful and/or offensive.

4.2 Data annotation

In the event dataset of 8000 texts, each text has a "is_humor" rating, which is a binary {0,1} value. This value is obtained by the organizers by asking annotators, "Is the intention of this text to be humorous? (0 or 1)". If the annotators answer yes, then they were asked to provide a rating of how humorous they found it, with a response in the range [1, 5]. The organizers took the majority label assigned by annotators, and the average of the ratings. Notably, they also allowed annotators to label a text as intended to be humorous (e.g. due to its content or structure) but also to give "I don't get it" as a rating. In this case, the humor rating for this annotator is 0. The organizers want to represent the subjectivity of humor appreciation with a controversy score. This examines the variance in humor ratings for each text. If the variance of a text was higher than the median variance of all texts, they then labelled the humor of the text as controversial. Prediction of this value is also a binary classification task.

The data also has two more labels. The annotators were asked, to give a binary label if they found the text to be generally offensive {0,1} and if they found it to be offensive then a rating with a response in the range [1, 5]. Offensive was defined as meaning that the text targets a person or group simply for belonging to a specific group, and asked users if they think that a significant number of people would find this offensive. The organizers saw much more variety in the offensiveness ratings, therefore they calculate an offensiveness score for each text. In this case, they consider the ratings [1-5], and also consider a no rating to be 0.

4.2.1 Custom Dataset Annotation

For our own datasets, we wanted to avoid having to manually label 200,000 individual texts, and instead relied on metadata to produce labels. As Reddit posts are voted on by the community, and given the communities these texts are from, we could create educated assumptions as to what metadata would constitute a positive or negative humor text. We worked with the assumption that all positive examples would come from the /r/jokes community, and all negative examples would come from the /r/worldnews or /r/reddit.com communities, due to the nature of the submissions. Inevitably, the scraped data included spam and low quality submissions, so we then performed some basic filtering to ensure quality. We found that by filtering on upvote ratios and the number of points a submission received, we could ensure that our custom dataset was of decent quality. After experimenting with various parameters, we found that requiring a minimum of 3 points and a ratio of 75% was a good balance between the quality of the text and dataset size.

The 200k Short Texts for Humor Detection dataset was also filtered by the dataset author to ensure quality. After the texts were collected, these were then filtered to only include texts with lengths varying between 30 and 100 characters, and with word lengths of between 10 and 18. These steps were taken to ensure that the sentence length distributions remained similar between the two classes, and could not be used to infer whether a text is a headline or a joke. Further preprocessing ensured that the case of all the texts followed

standard sentence case. 100,000 texts from each preliminary dataset were sampled to form a final dataset of 200,000 texts.

4.3 Data preprocessing

We do all of our work for this project using Python on Google Colab. For pre-processing, we use the HuggingFace tokenizers that correspond to the models we will use for training. We implement all models using PyTorch.

4.4 Data Insights

This section explores many of the characteristics of the competition dataset.

4.4.1 Data Statistics

Our objective here is to explore the fundamental characteristics of the text data we have. We analyze data points such as text lengths and frequently occurring words.

We parse the data and remove the stopwords using the NLTK library for English stopwords. We then plot the common unigrams terms in Figure 1. We find that words like **"like, people, wife, etc."** are words which occur with high frequency in the texts. At a high level these appear to be common joke topics/themes.

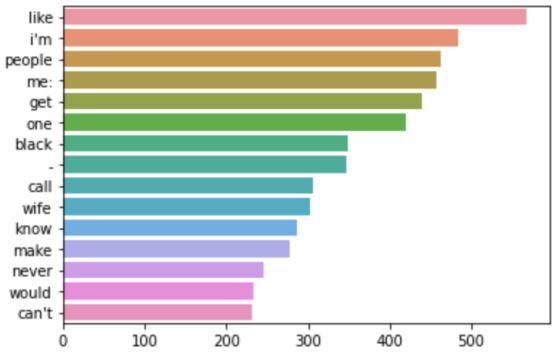

Figure 1: Most common unigrams

The trigrams plot shown in Figure 2, can be seen to have a lot of common joke lines such as **"how do you...", "what do you...", etc.**.

We also plot the distribution of text lengths for our own compiled dataset, the 200k humor dataset, and the competition dataset in order to ensure they follow a similar distribution. This can be seen in Figures 3, 4, and 5. We notice that the competition dataset is far less normally distributed than our own dataset or the 200k humor dataset, but the average length and the skew of the distribution towards shorter texts remain similar.

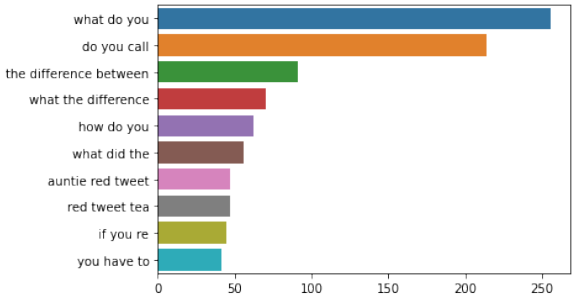

Figure 2: Most common trigrams

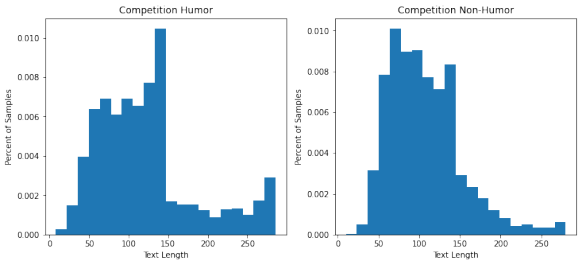

Figure 3: Text length distribution (in characters) for our competition dataset

4.4.2 Named Entity Recognition

Named entity recognition (NER) is an information extraction method in which entities that are present in the text are classified into predefined entity types like "Person", "Place", "Organization", etc. Our objective of using NER is so that we can gain insights into the types of entities present in the competition dataset. For this analysis we use the *"en_core_web_sm"* NER model from the spaCy[4] library.

We run NER on our dataset, then plot and visualize the entity frequencies as shown in Figure 6. We observe that the top categories of data are **DATE, PERSON and CARDINAL**. We evaluate the entities identified in each of the categories to further understand the data.

In Figure 7, we visualize the most common tokens per entity and observe that most jokes have a reference to dates in the form of **today, tomorrow, one day, etc.** We presume that some of these jokes have a temporal aspect to lay the foundation of the joke. The extracted sentences were also obtained from news headlines which could also explain the observation.

We plot some more of these top NER categories in Figures 8, 9 and 10. It shows that potentially the topic or theme of a joke is around a person (e.g. Jesus, God, etc.), an organization or institu-

[4]https://spacy.io/

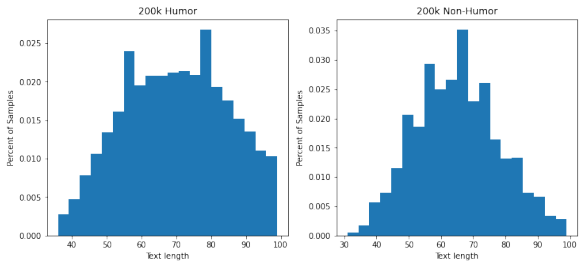

Figure 4: Text length distribution (in characters) for the 200k humor dataset

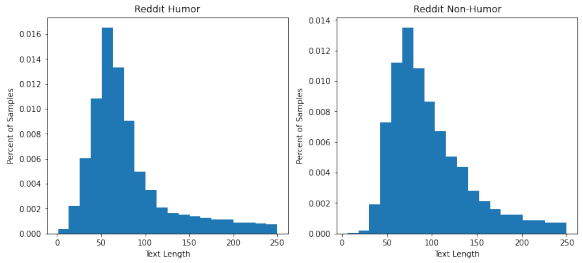

Figure 5: Text length distribution (in characters) for our Reddit dataset

tion (e.g. the US congress, McDonald's, etc.) or a NORP, which are Nationalities or religious or political groups (e.g. Mexicans, Muslims, etc.). This analysis helps us better understand our data, and the kind of datasets we would require to help improve for pretraining tasks. Based on our analysis, most of the sentences can be inferred to be a tweet or short review of an person, organization or group.

5 Baselines

Our dataset consists of four tasks, containing both binary classification and regression tasks. The competition itself does not provide any sort of baselines. For simple baselines on our development set, we include a constant majority class baseline for both classification task, and we use a constant average (over the training set) as the baseline for both regression tasks. For more advanced baselines on our internal development set, we use DistilBERT models trained individually for each of the four tasks. The hyperparameters that we use for this baseline (and the rest of our models) are: learning rate of 5e-5, number of warmup steps 500, weight decay 0.01, 3 epochs, and a batch size of 10 (to fit in the Colab GPU memory). We selected these hyperparameters by performing a small search over handpicked possible hyperparameter settings, using the score on the internal development set to select the best settings.

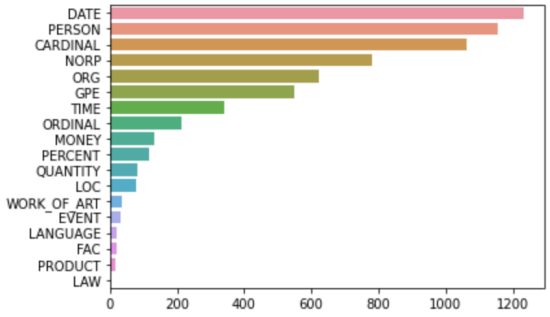

Figure 6: Top NER categories by frequency

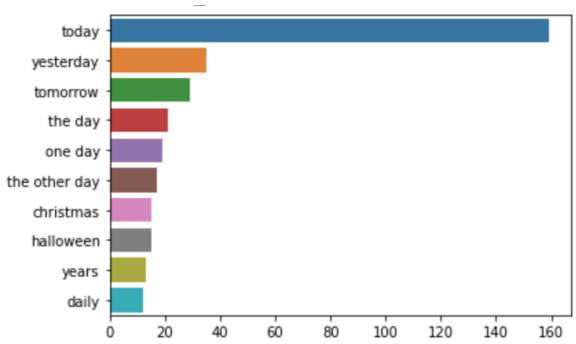

Figure 7: Top Date categories by frequency

For more advanced baselines on our test set (official development set), we include the top-3 approaches on the competition leaderboard as of the submission of this report. These baselines are included in Tables 1 and 2. The details about our test, train and dev set are covered in our previous section 4. We discuss each of these models in detail in the following section 6.

6 Your approach

Generally, we use the transfer learning paradigm that has come to dominate most NLP tasks. Specifically, we use the HuggingFace Transformers library[5], along with their large collection of pretrained language models, as a starting point. Then, we finetune these transformers on our training dataset (7200 examples, part of official competition training set). As previously mentioned, we use Python and PyTorch on Google Colab GPUs to train our models. To get the models to work on Colab, we had to use a relatively small batch size, as mentioned in Section 5. We use the same hyperparameters for training all of our models as those mentioned in Section 5. The Readme in our included repository outlines which Python files and

[5]https://huggingface.co/transformers

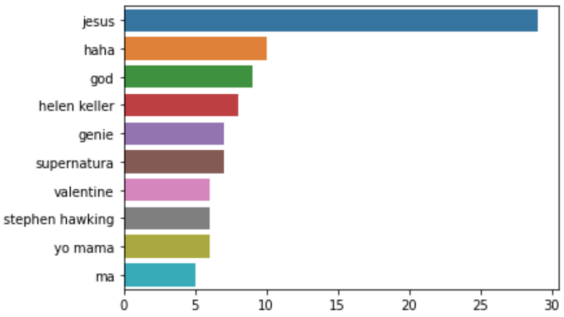

Figure 8: Evaluating the PERSON category from the NER response by frequency

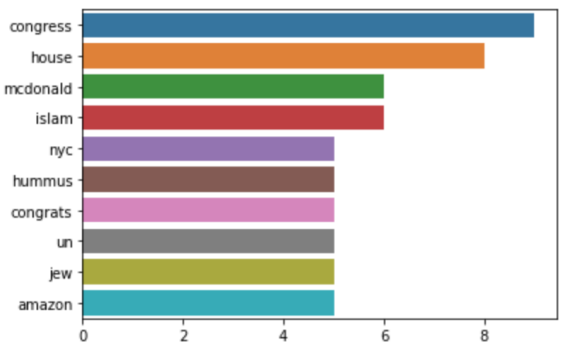

Figure 9: Evaluating the ORG from the NER response by frequency

Colab notebooks are associated with which models.

We evaluate our models first on an internal development dataset (800 examples, part of official training set) for model selection and hyperparameter tuning (Table 1). Then, we evaluate on the held-out official competition development set (competition test set will not be released until after project deadline) by submitting our predictions to the competition leaderboard for scoring (Table 2).

Building on this basic paradigm, we experiment in three different ways with the goal of improving model performance. We experiment **(1)** using various datasets for intermediate finetuning, **(2)** training multitask models to predict all labels simultaneously, and **(3)** ensembling predictions using different pretrained language models as starting points.

6.1 Intermediate Finetuning

We tried using intermediate finetuning (+IF) on larger related datasets for humor detection and offensive language detection in the hope that these larger datasets would provide a better starting point for training on the competition data, which is relatively small at just 8000 texts.

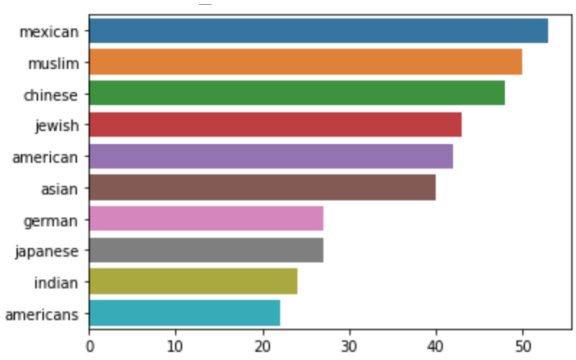

Figure 10: Evaluating the Person NORP from the NER response by frequency

We accomplish this intermediate finetuning in the same manner as the previously described basic transfer learning setup, and then we perform an additional transfer from the intermediate task to the competition task. For intermediate tasks we try using the two offensive language identification datasets previously mentioned, and for humor we tried using the 200k humor dataset (IF=200k) and the Reddit dataset we collected (IF=Reddit).

We also try using ColBERT (Annamoradnejad, 2020), a pretrained BERT model that has been finetuned for humor prediction, as a starting point for intermediate finetuning (ColBERT Humor +IF=200k) and as a pretrained language model for the standard transfer approach (ColBERT Humor). Their work proposes an approach that uses BERT to generate tokens and sentence embedding for texts. It sends embedding outputs as input to a two-layered neural network that predicts the target value. Following their approach, on the dev set for the event, we obtain an Humor F1 of 0.9185 and accuracy of 0.893 as show in Table 2. We then use ColBERT with IF on a 200k dataset[6] of formal short texts (100k positive, 100k negative), and we observe that the performance has a drop in both Humor F1 to 0.9047 and Accuracy to 0.88. This is counter-intuitive as IF should help us imrpove our performance. Our postmortem analysis on why this happens, leads us to identofy some variations in the kind of data the 2 humor dataset use. We discuss this in detail in our Error Analysis section.

[6]https://www.kaggle.com/moradnejad/200k-short-texts-for-humor-detection

6.2 Multitask Models

Initially, we train one model for each task or subtask. We also try training one model to learn to predict the labels associated with all four tasks or subtasks at the same time. To accomplish this, we attach four different heads on top of the final transformer outputs. Each prediction head consists of two fully-connected feed-forward layers matching the dimensionality of the transformer layers used by the pretrained language model, and an output layer that produces a single regression score or binary probabilities depending on the task.

6.3 Ensembling Model Predictions

We did most of our development with DistilBERT (unless otherwise specified) because it is relatively fast to train and run, allowing us to iterate more rapidly. We hypothesized that different pretrained language models would have different strengths and weaknesses when finetuned due to the different pretraining data used and the different model architectures. By ensembling (+Ensemble) many language models together, we might then counterbalance the weaknesses of individual models to improve overall performance.

Ultimately, we experimented with 7 model variants: "distilbert-base-uncased", "distilroberta-base", "bert-base-uncased", "albert-base-v2", "roberta-base", "bert-large-uncased-whole-word-masking", and "roberta-large" pretrained language models from the HuggingFace Transformers library. To get the predictions for each model, we average together the predictions from 5 different random restarts (RR) to mitigate the effect of variance induced by the random initialization. To ensemble the different models together, we simply averaged the predictions from each model together to form the final predictions, taking the argmax of the averaged probabilities for classification tasks.

For a slightly more advanced ensembling method, for each task we select the models to average together by trying all possible combinations and selecting the combination that leads to the best performance on our development dataset (+Ensemble-Best). Then, we use the same model combinations to generate predictions to submit to the competition leaderboard.

6.4 Results

Looking at the results on our development set in Table 1, we see that we easily outperform the simple "common-sense" baselines that we outlined in Section 5. We also find that intermediate finetuning on other larger datasets for humor and offense identification do not seem to benefit performance on our primary task, as measured against the baseline DistilBERT models trained individually for each task. Next, multitask learning seems to benefit performance for some tasks, such as humor rating prediction and humor controversy prediction, but may decrease performance slightly on offense rating prediction. Finally, we find that ensembling random restarts of the same model seems to reduce the effects of variance, leading to better results on average. Similarly, ensembling different pretrained language models together leads to an increase in performance, suggesting that these models complement each other by mitigating other models' weaknesses.

In Table 2, we see that these trends are largely replicated on the official competition development dataset (which we do not use for model selection or hyperparameter tuning). Furthermore, we see that in comparison to the top 3 submissions (other than us) on the leaderboard for each task, our methods outperform or perform competitively with the top approaches for each task, putting us at 2nd place in binary humor classification, 1st in humor rating regression, 1st in humor controversy classification, and 1st in offense rating regression, as of the time of this project submission.

7 Error analysis

One of the possible sources of confusion and bias in our model seemed to be centered around question marks. When a question mark was placed in the middle of a sentence, the model would often erroneously label it humorous regardless of actual content. Take for example the following mislabelled examples:

- Parenting Takes Mom and Dad.. So why do we humans insist on bucking nature? Successful parenting takes two – mom and dad. parenting family
- Want to know why he disappeared? These are the most common reasons men disappear from your life.

Approach	Humor F1 / Acc	Humor RMSE	Controversy F1 / Acc	Offense RMSE
Simple Baselines	0.763 / 0.617	0.567	0.0 / 0.500	0.980
Offense	– / –	–	– / –	0.534
Offense +IF	– / –	–	– / –	0.542
Humor	0.950 / 0.939	–	– / –	–
Humor +IF=200k	0.917 / 0.895	–	– / –	–
Humor +IF=Reddit	0.956 / 0.946	–	– / –	–
ColBERT Humor	0.920 / 0.900	–	– / –	–
ColBERT Humor +IF=200K	0.910 / 0.850	–	– / –	–
Individually trained	0.950 / 0.939	0.518	0.468 / 0.532	0.557
Multitask	0.948 / 0.936	0.498	0.538 / 0.550	0.564
Individually trained, 5 RR	0.952 / 0.941	0.507	0.537 / 0.540	0.544
Multitask, 5 RR	0.951 / 0.940	0.491	**0.574** / 0.559	0.547
RoBERTa-Large Multitask	**0.970 / 0.964**	0.483	0.567 / 0.554	0.518
Multitask +Ensemble=7LM	0.964 / 0.956	0.476	0.555 / 0.548	0.521
Multitask +Ensemble-Best=7LM	**0.970 / 0.964**	**0.472**	0.567 / **0.565**	**0.510**

Table 1: Internal Development Results

These were both labelled as being humorous by several models, and we believe that the presence of a question mark mid text is a large contributor towards that. When conducting a manual review of the data, we found that the vast majority of texts that contain a mid-text question mark are humorous due to their setup and punchline structure. Without balancing with negative examples with similar structures, the model can become reliant on punctuation structure rather than the actual relationship between the words.

Atypical or excessive punctuation in general seems to lead the model to believe a sample is humorous when it isn't, exclamation marks especially. Some other examples of non-humorous text being labelled humorous with excessive punctuation:

- IÆ'??m talking about ATMOSPHERIC FEEL, those fucking grain silos!!! That light!!
- The 1-2 punch of the Thong Episode plus the Spice Girls Episode last night was really something. When Maya takes a bite of the apple bong and throws it on the ground!!!!!!!!!!!!!!!!!!!

Another driver of error seems to be the actual source of the competition dataset. When we performed our deep dive analysis, we found that the vast majority of the negative examples seemed to be sourced from tweets. This can be seen in a striking manner with Figure 3, there's a sharp cutoff around 140 characters, what used to be the maximum length for a tweet. However none of the other datasets we found or compiled ourselves(for Humor tasks) contained information from twitter specifically, almost all relied heavily on news headlines instead. When pulling individual examples, we found that tweets tended to use more colloquial language, with a greater variety of punctuation, vocabulary, and capitalization when compared to news headlines. We've compiled some examples below:

Tweets

- TODAY IS THE DAY! Please join us at 6 p.m. for the kick-off of a year-long civic engagement series feat. Dr. Mindy Fullilove!
- ReneeÆ'??s Revenge yÆ'??all hahahahaha

News Headlines

- Obama's climate change legacy is impressive, imperfect and vulnerable
- Steelers coach incensed by headset situation at gillette stadium

As stated earlier, those with a large number of exclamation marks or those with question marks mid-text often confused the models, and negative examples were often only found in these tweets and almost never in news headlines.

Approach	Humor F1 / Acc	Humor RMSE	Controversy F1 / Acc	Offense RMSE
Simple Baselines	0.7745 / 0.6320	0.5722	0.0 / 0.5127	1.1962
Leaderboard[7] #1 (besides us)	**0.9505** / **0.9370**	0.5133	**0.6127** / 0.5759	0.5730
Leaderboard #2 (besides us)	0.9397 / 0.9230	0.5553	0.5370 / 0.5744	0.5738
Leaderboard #3 (besides us)	0.9320 / 0.9130	0.5766	0.5295 / 0.5585	0.6050
Offense	– / –	–	– / –	0.6045
Humor	0.9175 / 0.8930	0.5185	– / –	–
Humor +IF=200k	0.8880 / 0.8470	–	– / –	–
Humor +IF=Reddit	0.9250 / 0.9030	–	– / –	–
ColBERT Humor	0.9185 / 0.8970	2.6513	– / –	–
ColBERT Humor +IF=200K	0.9047 / 0.8800	2.5897	– / –	–
Individually trained	0.9175 / 0.8930	0.5185	0.5294 / 0.5443	0.6045
Multitask	0.9174 / 0.8920	0.5098	0.5284 / 0.5538	0.5982
RoBERTa-Large Multitask	0.9490 / 0.9340	**0.4739**	0.5581 / 0.5665	**0.4882**
Multitask +Ensemble=7LM	0.9348 / 0.9150	0.4820	0.5601 / 0.5775	0.5469
Multitask +Ensemble-Best=7LM	0.9490 / 0.9340	0.4848	0.5615 / **0.5823**	0.5239

Table 2: Competition Results (Official Development Set, Our Test Set)

One final potential driver of error were song lyrics and quotes. There were a proportionally large number of movie, song, and TV show quotes used in the dataset, by a rough estimate based on sampling, approximately 5% of the example fell in one of those categories. Our model was often able to differentiate between these quotes, though it was not something that was found in our own custom datasets. Moving forward, in order to improve accuracy, we will likely need to ensure our intermediate finetuning dataset has a similar distribution of quotes. Some examples below:

- "Only miss the sun when it starts to snow, only know you love her when you let her go." - Passenger.
- "Most misunderstandings in the world could be avoided if people would simply take the time to ask, 'What else could this mean?'" Shannon L. Adler. love u
- You can seek the advice of others, surround yourself with trusted advisors... but in the end, the decision is always yours, and yours alone.

After performing this deep dive analysis on our results, and seeing the various areas of where our model got confused, we believe that the primary reason our models did worse with the inclusion of extra datasets was due to the source of our data. The wider range of punctuation, capitalization, and vocabulary expressed in twitter posts was not well captured by utilizing news headlines as a negative source, and thus likely allow our model to use syntax and punctuation structure as a crutch.

8 Contributions of group members

- Gabriel Brookman: Worked on training individual models for the offensiveness task and prepared two of the datasets used for that task. He also did report writing.
- Akshay Gugnani: Worked on the data preprocessing and insights, initial experiment with baseline using sentiment & polarity, training individual models and pretrained models from prior work. He also did report writing.
- Nicholas Samoray: Worked on the compilation of custom 200k reddit dataset, testing effects of pretraining with various datasets and error analysis. He also did report writing.
- Brian Zylich: Worked on the multitask models, simple baselines, intermediate finetuning on humor dataset, ensembling different pretrained language models. He also did report writing.

9 Conclusion

Throughout the course of this project, we had the opportunity to gain experience with many of

the fundamental steps that go into building a system for an NLP task. We scraped and processed raw text into a cleaned dataset, we leveraged the transfer learning paradigm to take pretrained language models and finetune them for our downstream task, we modified the models to seek improvements in performance, and we analyzed the errors produced by our models to gain insights into how we could improve future performance.

One thing that we found quite surprising was that none of the intermediate finetuning that we did seemed to help. Normally in deep learning and NLP, more data tends to lead to better performance, but this was not the case in this instance. Upon analyzing our system's errors, we identified that this is likely due to the mismatch in how non-humorous examples are chosen in the competition dataset and the other datasets we sought to use. Namely, other datasets used text drawn from news articles, which are much easier to identify as non-humorous than random text drawn from social media.

This gives us a significant takeaway; we should more carefully inspect the nature and source of the text in a dataset prior to spending a significant amount of time trying to leverage additional training data that is very different in its style. Despite this hurdle, we managed to produce a system that is very competitive in the Hahackathon competition.

As of the end of day today, our team holds a position in the top 3 for each of the 4 tasks [8], and we hold the first rank for 3 of the 4 tasks, among 21 competing teams. As this competition continues through the end of January, we will likely take what we have learned and attempt to gather additional training data that matches up better with the type of data seen in the competition in the hope that this gives us a further competitive edge.

References

Annamoradnejad, I. (2020). Colbert: Using bert sentence embedding for humor detection. *arXiv preprint arXiv:2004.12765*.

Badlani, R., Asnani, N., and Rai, M. (2019). Disambiguating sentiment: An ensemble of humour, sarcasm, and hate speech features for sentiment classification. *W-NUT 2019*, page 337.

Cai, F. (2019). Does ai get the joke?

Dai, Z., Yang, Z., Yang, Y., Carbonell, J., Le, Q. V., and Salakhutdinov, R. (2019). Transformer-xl: Attentive language models beyond a fixed-length context. *arXiv preprint arXiv:1901.02860*.

Davidson, T., Warmsley, D., Macy, M., and Weber, I. (2017). Automated hate speech detection and the problem of offensive language. In *Proceedings of the 11th International AAAI Conference on Web and Social Media*, ICWSM '17, pages 512–515.

Devlin, J., Chang, M.-W., Lee, K., and Toutanova, K. (2018). Bert: Pre-training of deep bidirectional transformers for language understanding. *arXiv preprint arXiv:1810.04805*.

Donahue, D., Romanov, A., and Rumshisky, A. (2017). HumorHawk at SemEval-2017 task 6: Mixing meaning and sound for humor recognition. In *Proceedings of the 11th International Workshop on Semantic Evaluation (SemEval-2017)*, pages 98–102, Vancouver, Canada. Association for Computational Linguistics.

Hossain, N., Krumm, J., Gamon, M., and Kautz, H. (2020). Semeval-2020 task 7: Assessing humor in edited news headlines. *arXiv preprint arXiv:2008.00304*.

J. A. Meaney, e. a. (2020). Hahackathon: Detecting and rating humor and offense.

Liu, Y., Ott, M., Goyal, N., Du, J., Joshi, M., Chen, D., Levy, O., Lewis, M., Zettlemoyer, L., and Stoyanov, V. (2019). Roberta: A robustly optimized bert pretraining approach. *arXiv preprint arXiv:1907.11692*.

Mao, J. and Liu, W. (2019). A bert-based approach for automatic humor detection and scoring. In *IberLEF@ SEPLN*, pages 197–202.

Meaney, J. (2020). Crossing the line: Where do demographic variables fit into humor detection? In *Proceedings of the 58th Annual Meeting of the Association for Computational Linguistics: Student Research Workshop*, pages 176–181.

Morishita, T., Morio, G., Ozaki, H., and Miyoshi, T. (2020). Hitachi at SemEval-2020 task 7: Stacking at scale with heterogeneous language models for humour recognition. In *14th International Workshop on Semantic Evaluations (SemEval-2020)*.

Radford, A., Wu, J., Child, R., Luan, D., Amodei, D., and Sutskever, I. (2019). Language models are unsupervised multitask learners.

Weller, O. and Seppi, K. (2019). Humor detection: A transformer gets the last laugh. *arXiv preprint arXiv:1909.00252*.

Yang, D., Lavie, A., Dyer, C., and Hovy, E. (2015). Humor recognition and humor anchor extraction. In *Proceedings of the 2015 Conference on Empirical Methods in Natural Language Processing*, pages 2367–2376.

Yang, Z., Dai, Z., Yang, Y., Carbonell, J., Salakhutdinov, R. R., and Le, Q. V. (2019). Xlnet: Generalized autoregressive pretraining for language understanding. In *Advances in neural information processing systems*, pages 5753–5763.

[8] https://competitions.codalab.org/competitions/27446#results

Zampieri, M., Malmasi, S., Nakov, P., Rosenthal, S., Farra, N., and Kumar, R. (2019). SemEval-2019 task 6: Identifying and categorizing offensive language in social media (OffensEval). In *Proceedings of the 13th International Workshop on Semantic Evaluation*, pages 75–86, Minneapolis, Minnesota, USA. Association for Computational Linguistics.