That's What Who Said

Justin Minnion Chris Chavez Kelsey Fox

DSCI 691 NLP Group Project

Abstract

We sought to create a tool to identify speakers in multi-party dialogues using text-based features. We conducted five experiments using two pre-trained transformer-based models (DistilBERT and RoBERTa) to predict if the speaker of a line of dialogue from the television show, *The Office,* was either "Dwight" or "Not Dwight". Of the five experiments we conducted, none excelled at the identification task, which we benchmarked with a trivial classifier, which always guessed "Not Dwight", that had an 82.9% accuracy. The best performing was our second experimental model—*RoBERTa*—, with 82.5% accuracy, still below our benchmark. With further hyperparameter tuning and work toward reducing class imbalance, we believe our experiments could produce better results.

1 Experimental Design

This section introduces the two models we selected and explains our implementation and reasoning for each of the five experiments we conducted.

Overall, we tried to answer the following questions with our experiments. The models used to explore these questions are noted in parenthesis.

- Can we consistently identify a character only using their lines of dialogue? (All)
- Does the length of the dialogue lines affect our ability to make this identification? (Remove Short Lines)
- Are our transformer models more effective than strategic (random) guessing? (All)
- How do different tokenizers affect transformer model performance (RoBERTa vs DistilBERT)?
- How much does class imbalance affect our results, and can we reduce its impact? (Rebalance)
- Will our identifications be more accurate if we include script metadata (e.g., director and writers) for the episode? (Augmented)

1.1 Model Selection: Transformer-based Pre-trained Language Models

Our NLP task is best described as a binary sequence classification task, so we selected two pretrained autoencoding models from the HuggingFace hub: DistilBERT and RoBERTa, described below. Both models are based on Bidirectional Encoder Representations from Transformers (BERT), which is a well-known model for binary sequence classification NLP tasks due to its ability to capture long-range dependencies and contextual relationship with text sequences.

1.1.1 Hugging Face DistilBERT Model

The DistilBERT model (Sanh, Victor, Debut, Chaumond, & Wolf, 2020) is based on the popular autoencoding BERT transformer model. (Devlin, Chang, Lee, & Toutanova, 2019). It uses a distillation technique to compress the BERT model into a smaller model that is 60% faster and smaller than BERT. It does this by removing the token-type embeddings and pooler while keeping the rest of the architecture and by reducing the number of layers by a factor of two to produce a model with 40% fewer parameters that can achieve more than 95% of BERT's performance. (Sanh, 2019) This

speed and performance was a key reason for selecting this model.

For the tokenizer, we used the DistilBertTokenizerFast, which is identical to the BertTokenizerFast. It runs end-to-end tokenization, meaning it applies punctuation splitting and wordpiece. Wordpiece describes the process of breaking words into smaller units called subword tokens (or wordpieces) to capture more detailed information and better handle out-of-vocabulary cases and rare words. This model uses character-level byte-pair encoding (BPE), which differs from the other model we used, discussed in the next section. (HuggingFace, 2020) (Wu, 2016)

1.1.1 Hugging Face RoBERTa Model

The **Ro**bustly **o**ptimized **BERT** **a**pproach (RoBERTa) model (Liu, et al., 2019) was developed by Facebook and was also based on Google's BERT model, but it has two major differences from BERT: (1) it uses a byte-level byte-pair encoding (BPE) as a tokenizer and (2) a different pretraining scheme.

As a quick review, this byte-level BPE means that instead of using unicode characters as the base subword units, it uses bytes. This allows it to learn a subword vocabulary of 50K units that can encode any input text *without* introducing unknown or out-of-vocabulary tokens. This is a larger vocabulary than that of DistilBERT.

We used the RobertaTokenizerFast as our tokenizer with this model. (HuggingFace, n.d.)

1.2 Experimental Approach: Model Implementation Process

To answer our experimental questions, we conducted five different experiments. Four of the five experiments used DistilBERT and one used RoBERTa. Our process is described below.

1. Preprocess datasets using standard Python packages (pandas, nltk, sklearn) to standardize speaker names, filter speakers to the 10 most frequent, remove problematic episode dialogue, merge the episode metadata (writers and director) into our script dataframe for experiment five, and export the results as processed datasets.
2. Implement models to address our experimental questions. Implementing models followed a consistent process:
    a. Convert Pandas dataframe to HuggingFace dataset.
    b. Obtain train/test/validation splits.
    c. Tokenize and encode using appropriate HuggingFace model tokenizer.
    d. Create model from the pre-trained HuggingFace transformer (setup training arguments and evaluation metrics) and then run the HuggingFace Trainer.
    e. Evaluate the model.
    f. Review results (accuracy, F1 score, precision, and recall) and document conclusions.
3. Repeat sub steps 2a–f, modifying the model or our approach to try and improve results.

2 Materials

All materials are hosted on our github project page: https://github.com/Zu1uDe1ta/thats-what-who-said. The rest of this section summarizes this content.

2.1 Datasets

Our datasets came from Kaggle and are summarized below.

2.1.1 Transcript data (character dialogue)

Filename
The-Office-Lines-V4.csv

- Author: Kaggle user Nasir Khalid
- https://www.kaggle.com/datasets/nasirkhalid24/the-office-us-complete-dialoguetranscript.

2.1.1 Episode datasets

In addition to the Kaggle datasets below, we consulted Wikipedia to confirm episode counts and that episode writers and directors were correctly mapped after our data merging (Wikipedia, n.d.).

Filenames (two csv files)
the_office_epsisodes.csv, the_office_imdb.csv

- Author: Kaggle user Bill Cruise

- https://www.kaggle.com/datasets/bcruise/the-office-episodes-data.

2.2 Jupyter Notebooks

We completed this work with two jupyter notebooks: (1) **01_preprocess.ipynb**, and (2) **02_transformer_model.ipynb**. Notebook (1) includes exploratory data analysis (EDA) of episode and transcript data, which informed later decisions during model tuning (e.g., padded sequence length and minimum line length) along with all preprocessing performed to clean and prepare the dataset for the transformer models.

Notebook (2) contains all the code and documentation needed to reproduce each of our transformer models.

2.3 Models

The model files were too large to host on github but are in our final project zipfile. Note: running the **02_transformer_model.ipynb** notebook will save the models to your machine.

Documentation for the two base models can be accessed as follows:

- DistilBERT base model (uncased): https://huggingface.co/distilbert-base-uncased
- RoBERTa base model (case-sensitive): https://huggingface.co/roberta-base

3 Results and Discussion

The performance of our methods and models are summarized in **Table 3**. Recall that our goal is to predict the label (speaker) given a line of dialogue from the television series, *The Office*. We simplified this to a binary prediction, where our models would identify the speaker as either "Dwight" (positive class: 1) or "not Dwight" (negative class: 0). Before discussing the performance of each model, we summarize observations that influenced our model experiments.

3.1 Dataset Discussion

Looking at the transcript dataset (The-Office-Lines-V4.csv), we see that the top ten speakers in the show accounted for ~72% of all show dialogue (i.e., lines, or utterances) with the most lines spoken by Michael Scott (Steve Carrell) at 19.7%, followed by Dwight with 12.4%. This distribution is shown in **Figure 1**. For more information on the preprocessing of this dataset and other EDA, consult the **01_preprocess.ipynb** file.

The next important observation about the dataset is the class imbalance, shown in **Table 1**. As part of our experiments, we attempted to re-balance the data in our third model because class imbalance is a known challenge for NLP and ML models, particularly for binary classification (Or, 2023). This is because class imbalance often leads to poor model performance on minority classes, and so far, transformer models like BERT have not solved this problem despite their strong performance compared to earlier models, like neural networks and traditional models (Henning, Beluch, Fraser, & Friedrich, 2023).

Experiment Number/Name (Transformer Model)	Positive Class	Negative Class	Class Imbalance Ratio
Experiments (1), (2), (5)	6 ,752	32,668	1:4.8
(3) Rebalance (DistilBERT)	13,504	16,334	1:1.2
(4) Remove Short Lines (DistilBERT)	6,752	32,668	*1:4.8

Table 1. The re-balanced class representation (experiment 3) was only applied to the training set. *Note: After modifying the dataset to discard lines with low word count (≤ 5), its class imbalance ratio was 1:4.7, a slight reduction, which is why it is listed separately.

We considered two approaches to solve this problem: (1) undersampling the negative class and (2) oversampling the positive class. (see notebook **02_transformer_model.ipynb**, Section 3 for more detail on this process.) For minimal complexity, we applied a 2x oversample and 2x undersample.

The word count distribution for each line was

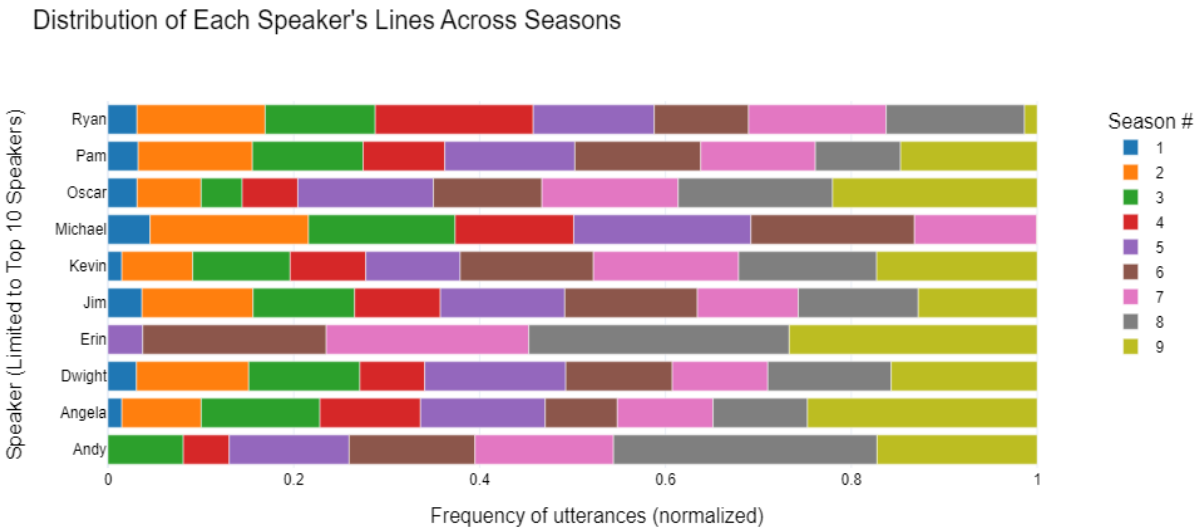

Figure 2. Distribution of each of the top ten speakers across the nine seasons of the television show, *The Office*. Dwight was one character who appeared across all seasons. To make sure we had enough data for our predictions, we selected this character as our target for our binary classification predictions.

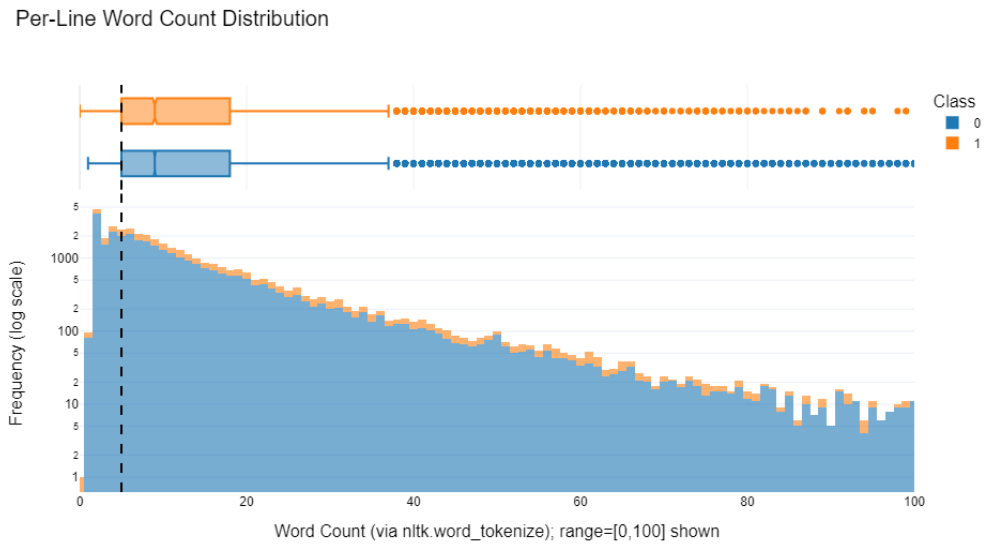

Figure 1. The per-line word count distribution. The black dashed line is drawn at word count equal to 5, which we used as our cut-off point, meaning that any lines with five or fewer words were dropped from our dataset. This modified dataset was then used in our fourth model experiment (Remove Short Lines).

the final observation that came from EDA that affected our model development, shown in **Figure 2.** As an additional experiment, we removed "short lines" from the dataset, as we assume very short lines ($N \leq 5$) may be too challenging to attribute to a specific person. Overall, this filtering reduced our dataset by 30%, which changed the split size for our train/test/validation datasets for this experiment (Remove Short Lines).

3.2 Model Results and Discussion

For all model experiments, we tried to maintain consistent encoding settings and train/test/validation splits for best experimental practices (see **02_transformer_model.ipynb**). This notebook also includes a sanity check we performed for each experiment, where we fed in two control lines of dialogue: (1) "Assistant to the regional manager of beets, Mose and mother on the farm"—a line that features strong Dwight-isms, and (2) "My name is Michael Scott, paper is my business"—a line explicitly telling us the speaker is not Dwight.

We evaluated each of our models according to their accuracy, F1, precision, and recall scores, shown in **Table 3** and **Table 2**. As **Table 2** makes very clear, none of the models outperformed our trivial classifier's accuracy score (82.9%), likely because of our class imbalance problem.

The class imbalance heavily favored "Not Dwight" since the ratio of lines spoken by "Dwight" vs. "not Dwight" was about 1:4.8, respectively. In fact, our attempt to correct this issue with the Rebalanced model had the lowest accuracy score (74.8%) for the validation results.

We considered the importance of precision and recall as equal with precision having a slight edge. This is because higher precision suggests fewer false positives, more true positives, or both. For our theoretical application (using this labeling to improve closed captioning) that would mean labeling DWIGHT in the captioning with a line that viewers do not see him speaking, which would be disorienting and confusing. On the other hand, higher recall suggests fewer false negatives, more true positives, or both. Since our current process is only the first step and does not deal with labeling other speakers, mislabeling a line spoken by Dwight as Not Dwight would likely signal the need for further analysis. We also think future work, beyond this project, could apply other modeling techniques (boosting) to these difficult-to-classify cases. Essentially, our argument is that false positives will always be problematic while false negatives may not be if downstream modeling can address it, thus precision is a slightly more important metric. The F1 score is also an important metric since it encapsulates both precision and recall.

We conclude this section with some brief notes on each model's performance. For a deeper discussion on each model and its results, consult the **02_transformer_model.ipynb** notebook.

3.2.1 Basic Transformer—DistilBERT

Overall, this model was unsuccessful. Its performance metrics fell significantly in the test and validation sets, for example F1 Score went from 87.5% for the testing set down to 33.0% and 32.2% for the testing and validation sets, respectively.

3.2.2 Modified Approach—Different Pretrained Language Model (PLM): RoBERTa

This experiment was done to see if a different model could produce better results. We thought the tokenization approach may improve our results. Ultimately, this model produced the highest accuracy score (82.5%), but it still fell short of our trivial classifier benchmark. The training time was also increased by a factor of ~4.6, which was expected, but the improvement did not seem worth the result.

3.2.3 Modified Approach—Re-balance Data: DistilBERT

To try and improve the class imbalance issue, we applied our oversampling of the positive class and undersampling of the negative class technique, as discussed earlier in this paper. The validation accuracy (75.0%) unfortunately signaled a step backward and the worst performance so far; however, the validation F1 score did show an improvement of 0.06. Overall, the model still did not pass our benchmark and was unsuccessful.

3.2.4 Modified Approach—Remove Short Lines: DistilBERT

We conducted another experiment to see if the model would improve if we removed lines that only had five words or fewer. This did change the test/train/split to 70/15/15 since we reduced the dataset by 30%. It had the highest training accuracy score, but still could not beat the benchmark.

Overall, this model achieved better results for every metric when compared to our base model and was an approximate tie with the RoBERTa model. The downside was losing a significant amount of the transcript data, and the new problem that introduced since those shorter lines would still need to eventually be labeled for our theoretical application (improving closed captioning transcripts to include speakers for all dialogue).

3.2.5 Modified Approach—Augmented Vocab: DistilBERT

For our last experiment, we updated our vocabulary, which had previously been lines of dialogue to now include the director's name and the lead writer's name in addition to the line, which we appended to the end of the dialogue line. We acknowledge this solution does mangle the original transcript data and could be improved.

Overall, the results suggested there was potential for improvement with hyperparameter tuning, however, its validation accuracy score did not beat the RoBERTa model's performance

Experiment Name: (Transformer Model)	Accuracy Train / Test / Valid	F1 Score Train / Test / Valid	Precision Train / Test / Valid	Recall Train / Test / Valid	Fine-Tuning Time
(1) Basic Transformer (DistilBERT)	0.961 / 0.810 / 0.806	0.875 / 0.330 / 0.322	0.978 / 0.417 / 0.402	0.791 / 0.274 / 0.268	0d 0h 09m 48s
(2) Different PLM (RoBERTa)	0.917 / 0.824 / 0.825	0.703 / 0.314 / 0.324	0.909 / 0.474 / 0.480	0.573 / 0.235 / 0.245	0d 0h 45m 23s
(3) Rebalance Data (DistilBERT)	0.956 / 0.747 / 0.748	0.951 / 0.375 / 0.373	0.969 / 0.325 / 0.325	0.934 / 0.443 / 0.437	0d 0h 09m 42s
(4) Remove Short Lines (DistilBERT)	0.989 / 0.811 / 0.814	0.969 / 0.372 / 0.384	0.988 / 0.442 / 0.456	0.951 / 0.321 / 0.332	0d 0h 08m 47s
(5) Augmented Vocab (DistilBERT)	0.977 / 0.805 / 0.807	0.929 / 0.328 / 0.334	0.977 / 0.400 / 0.409	0.885 / 0.278 / 0.283	0d 0h 09m 58s
Trivial Classifier (always call “negative”)	- / 0.829 / 0.829	-	-	-	-
*Trivial Classifier (Re-Balance Data only)	0.547 / 0.829 / 0.829	-	-	-	-

Table 3. Table of all model performance metrics. This includes a “trivial classifier” benchmark, which we used to make sure our basic transformer model test and validation accuracies were more effective than guessing “not Dwight” for each line of dialogue. This was done because of the class imbalance in this dataset (1:4.8 positive (Dwight) to negative (not Dwight) ratio, respectively). *Note: The second trivial classifier benchmark has a modified value of 0.547 (but only for the training data) because we modified the training dataset to account for the class imbalance. This was only applied for this specific model and experimental method. See section 3.1 and 3.2.3 for more information.

365

Experiment Name: (Transformer Model)	Accuracy	F1 Score	Precision	Recall
(1) Basic Transformer (DistilBERT)	0.806	0.322	0.402	0.268
(2) Different PLM (RoBERTa)	**0.825**	0.324	**0.480**	0.245
(3) Rebalance (DistilBERT)	0.748	0.373	0.325	**0.437**
(4) Remove Short Lines (DistilBERT)	0.814	**0.384**	0.456	0.332
(5) Augmented Vocab (DistilBERT)	0.807	0.334	0.409	0.283
Trivial Classifier (always call “negative”)	0.829	-	-	-

Table 2. Validation model performance summary table easier comparison and evaluation of model performance of all experiments and models. Bold emphasis used to highlight the highest scores for each metric. Unfortunately, none of our models beat the trivial classifier (our random guesser), so none of the models succeeded at the task.

References

Devlin, J., Chang, M.-W., Lee, K., & Toutanova, K. (2019). BERT: Pre-training of Deep Bidirectional Transformers for Language Understanding.

Henning, S., Beluch, W., Fraser, A., & Friedrich, A. (2023). A Survey of Methods for Addressing Class Imbalance in Deep-Learning Based Natural Language Processing. *Proceedings of the 17th Conference of the Eurpoean Chapter of the Association for Computational Linguistics*, (pp. 523-540).

HuggingFace. (2020). *DistilBERT*. Retrieved from HuggingFace.co: https://huggingface.co/transformers/v3.0.2/model_doc/distilbert.html

HuggingFace. (n.d.). *Tokenizer*. Retrieved from HuggingFace.co: https://huggingface.co/transformers/v3.0.2/main_classes/tokenizer.html

Liu, Y., Ott, M., Goyal, N., Du, J., Joshi, M., Chen, D., . . . Stoyanov, V. (2019). RoBERTa: A Robustly Optimized BERT Pretraining Approach.

Or, B. (2023, January 03). *Solving The Class Imbalance Problem*. Retrieved from Medium: https://towardsdatascience.com/solving-the-class-imbalance-problem-58cb926b5a0f#:~:text=Class%20imbalance%20is%20a%20common%20problem%20in%20machine%20learning%20that,can%20negatively%20impact%20its%20performance.

Sanh, V. (2019, August 28). *Smaller, faster, cheaper, lighter: Introducing DistilBERT, a distilled version of BERT*. Retrieved from Medium: https://medium.com/huggingface/distilbert-8cf3380435b5

Sanh, Victor, Debut, L., Chaumond, J., & Wolf, T. (2020). DistilBERT, a distilled version of BERT: smaller, faster, cheaper and lighter.

Wikipedia. (n.d.). *List of The Office (American TV series) episodes*. Retrieved from Wikipedia: https://en.wikipedia.org/wiki/List_of_The_Office_(American_TV_series)_episodes

Wu, Y. S. (2016). Google's Neural Machine Translation System: Bridging the Gap between Human and Machine Translation. *ArXiv*.