Marvin: Stylistic Language Editor Progress Report

Vivek Aithal, Priyam Srivastava, Daniel McAndrew

March 2021

We are currently working on using contemporary natural language processing (NLP) methods to create an intuitive AI powered tool for writing. In specific, we want to understand how to generate and edit natural language documents to reflect certain styles. Several theories of language suggest that the meaning of utterances is socially constructed and understood through social contexts (Bakhtin and Holquist 1981). Many factors (like a speaker's gender, race, age, occupation, social status, relationship with their listener, current emotional state etc.) are known to influence how language is structured and used to refer to concepts. One important factor that drives writing style is the domain, since many domains have specific stylistic restraints on how language ought to be structured. Some of these domain-specific restraints can pose significant challenges to new writers. We are building algorithmic tools that can aid users in such dilemmas by suggesting edits and additions to meet stylistic constraints. Modern NLP has provided machine language understanding and generation capabilities which are now the frontier of such assistance, going beyond the suggestion of hard-coded "best" practice heuristics as previous tools have done. Our project aims to leverage a combination of recent innovations such as pre-trained transformer models like BERT (Devlin et al. 2018), perhaps with our own enhancements, to provide AI-powered stylistic writing assistance. We envision the tool that we are developing to adhere to a machine-in-the-loop framework, where the writing is performed largely by human users, but is aided by algorithmic suggestions.

Links

We have created a site describing the project on Berkeley's School of Information website. We also have been developing the project in this Github repository. We're tracking our tasks and general progress on this this Github Kanban Board.

Goals

After going through the state of the art in language style transfer, and tools that aid such edits, we have narrowed down our goals to two things.

- A web application for writing documents that enables users to :
  - View the style statistics of a particular sentence, understand which tokens contribute most to the style and help users rewrite themselves.
  - Select a sentence and get suggestions for how that sentence could be rewritten in a particular style of choice (or multiple styles).
  - View suggestions for sentence auto-complete in a particular style
- Report the cross correlations of how changing one style affects all the other styles similar to the work done by Kang and Hovy 2019 in xSLUE

Exploratory Work - ML

We have spent time reading through the literature related to the problems that we want our tool to address and have tried a number of approaches with varying degrees of success. Here we describe some of the related research and our experiments using some of these approaches so far. We have been conducting our experiments primarily on stylistic language datasets which have been compiled in Kang and Hovy 2019.

- Self Attention and BERT

  We implemented a Self-attentive Sentence Embedding model for text classification (Lin et al. 2017). This approach introduced a self-attention mechanism that has been very influential in the development of transformers (Vaswani et al. 2017) and the models upon which they are based such as BERT.

We wanted to use a version of this self-attentive model to classify text as belonging to a certain style and provide users with a visualization of the attention mechanism along with the text's score for each of the style classes. We want users to be able to understand not just which style the model predicted for their text, but also how confident the model is and which tokens contributed most to the style. This model predates transformers and BERT and performs less well than these more recent approaches on many NLP tasks, however, we considered this model because we think that it may be easier for a human to visualize and interpret how the model is arriving at its prediction.

We implemented a version of this model using PyTorch, where a word embedding layer was initialized with weights from GloVe vectors (Pennington, Socher, and Manning 2014). We then trained this model on the Stanford Politeness dataset (Danescu-Niculescu-Mizil et al. 2013) to classify text as either polite or impolite. In our experiments we found that the model was almost achieving a performance on this classification task as that seen in Kang and Hovy 2019, but it was still not quite achieving the accuracy that we wanted. We were able to extract the attention and visualize it though, which was our main goal for using this model. Some of these visualiations made sense, but there was still some shortcomings. On some of our other datasets, such as the Sentiment Treebank (Socher et al. 2013) and the ShortHumor dataset compiled by Kang and Hovy 2019, this model performed much better, so we still may ultimately include this model in our final application depending on how it compares to come of the other approaches.

Because the self-attentive model was not working as well as we wanted initially, we decided to try using BERT models as well. We believed that a BERT based model would perform better on the classification task, but its attention mechanisms might be harder to visualize and interpret. We used a pretrained DistilBERT model (Sanh et al. 2019) from Hugging Face's Transformers library (Wolf et al. 2020) which had been pretrained on the Sentiment Treebank. We then wrote a pipeline to finetune this model for our datasets.

This DistilBERT model did perform better on the politeness, humor, and sentiment classification tasks than the self-attentive approach. Our best experiments with this model so far have achieved around 70% validation accuracy on the politeness task, 80% validation accuracy on the humor task, and over 90% on positive/negative sentiment, which are promising results and close to the state of the art. These experiments were more exploratory in nature and we have not yet made an effort to tune hyperparamers and optmize model performance. This is future work that we intend to do. Also, as expected, it was initially less clear how to best visualize the attention compared to the self-attentive model. However, after reading through some of the literature on this topic and inspecting other similar tools like BertViz (Vig 2019), we were able to create similar attention visualizations as with the previous model considered. In the figures section, we show some results of this attention visualization along with the model's predictions on some unseen test sentences.

- xSLUE

  Cross-Style Language Understanding and Evaluation (xSLUE Kang and Hovy 2019) provides a benchmark corpus for understanding language styles, with a focus on how styles can be dependent on one another and are coupled to content and domain to varying degrees. It contains 15 different styles and 23 classification tasks related to these styles. We have been using some of these styles and datasets for the work that we have done so far and have been comparing our work to the benchmarks set by xSLUE. Kang and Hovy 2019 also provides analysis of the associations between different dimensions of style by measuring the correlation of different styles. They do this by using all of the trained classifiers to predict on a new set of texts from tweets and then quantifying the co-occurence. They also sample text from different domains and take a similar approach to understand the stylistic diversity of the selected domains. We are also interested in understanding fine-grained styles like those from xSLUE and also the macro-styles of different domains which are in some senses distributions over more fine-grained aspects of style. We have been inspired by XSLUE and intend to perform an analysis of cross-style dependencies for stylistic language editing and generation in a similar way to how they did it for classification.

- Autoencoder This paper by Shen et al. 2017 demonstrates a classic approach of solving the style transfer problem in text. They use an encoder that takes a sentence and its original style as input and converts it into a paraphrase/style-independent representation. This style-independent content is then used as input by a generator that generates the sentence in the desired style. Like Krishna, Wieting, and Iyyer 2020, this paper also tries to strip away the style of the given data point and understand the true content. Then as the final step, apply new style to this content. The key difference here is the direct application of auto-encoder where they introduce a latent content variable z, and latent style variable y for each style. Both the latent variables are being learnt explicitly by the auto-encoder using neural networks for $P(z \mid x, y)$ and $P(x \mid y, z)$. They also add a constraint of consistency in the distribution (cross-alignment) by using the two styles from the same distribution in the input. By using the corpora from same source, they add the constraint that sentences from one style should be similar to the other style from the same population.
  We tested it with positive and negative sentiments as two styles. Some examples from the model:
  positive to negative:
  sentence: excellent chinese and superb service
  transfer: terrible food and poor customer service

  negative to positive:
  sentence: i am really disappointed
  transfer: i am feeling great

- TextSETTR This paper by Riley et al. 2020 paper uses a label-free approach of solving the problem of style transfer. The approach taken by Shen et al. 2017 and Krishna, Wieting, and Iyyer 2020 uses non-parallel data but still each corpus has a style associated to it. They use style as a label. However, this paper completely eliminates the need for labels. It exploits the fact that there is a connection in style between the adjacent sentence (unlabeled). While training, they use the adjacent sentences(latter sentence with noise) to learn a Style Extractor and an Encoder. As the next step, Encoder conditioned on the Style Extractor tries to recreate the sentence thus learning a decoder. The learned Style Extractor enables us to tune the weights of the style during inference.

We believe that is a very significant contribution in the field of Style Transfer in text. "Tunable Inference" could be one of the key feature of our application. This will allow the user to decide the weights of each style and get a coherent sentence based on their custom-designed styles.
We are still exploring and implementing this technique. We are hoping to have some results for it soon.

- Plug and Play Language Models (PPLM)

  This is a very exciting paper by Dathathri et al. 2020 from Uber, which uses a discriminator model to guide the language model's (LM) generation. They achieve this by a forward and backward pass of the text generated until step t, and the gradients from these attribute models tweak the LM's hideden activations to result in the next token that matches the attribute discriminator. The language model does not require any training and can hence be a generic pretrained model such as GPT-2, and the discriminator models are far easier to train (with $\sim$ 5 orders of magnitude fewer parameters). But our experimentation showed that while PPLM works very well given a prompt, it fares quite poorly when it has to rewrite a given sentence. We concluded that this approach does not serve our current goals and could potentially be used as a stylistic "auto-complete" feature (without having to fine-tune a separate LM for each style class).

- Tag and Generate

  In this paper, Madaan et al. 2020 perform style transfer in two steps. First, they use a tagger-model to tag all the slots that could be replaced, deleted or inserted in a given sentence to convert the sentence to a desired style, and then they use a second model to fill in these tagged slots. They do not need a parallel dataset to train, since the tagger is first trained on a synthetically created dataset using n-gram tf-idf to identify stylistic phrases. The generator is trained on the earlier created tagged intermediate sentence. We were initially very excited about this approach, but soon realized that this is limited by its construction. It works well when the only modification needs is one of addition, deletion or replacement. This approach performs poorly if we need a complete rewrite of the text.

Input	Tagged	Output
send me the text files.	[TAG] send me the text files.	could you send me the text files.
look into this issue.	look [TAG] into this issue.	look forward to hearing from you into this issue.
pass me the salt.	[TAG] pass me the salt.	could you please pass me the salt.
I hate this.	I hate [TAG] this.	I hate to help you with this.
what the hell?	what the hell?	what the hell?

Table 1: Some Tag and Generate examples we experimented with to transfer the ”Politeness” style, that highlights the performance and the inadequacies of the system

- Reformulating Unsupervised Style Transfer as Paraphrase Generation

  In this approach, Krishna, Wieting, and Iyyer 2020 rewrite a sentence in one style to another style, by using an intermediate paraphrase representation. The unique element of this paper is that they achieve state of the art results on style transfer without needing parallel datasets in those styles. They achieve this by first generating a “diverse” paraphrase of the input sentence thus stripping the sentence of its style. Then, the paraphrased sentence is rewritten in the style of choice. Once the paraphrase model is trained (separately on a parallel paraphrase dataset), we can generate that to create pseudo-gold parallel data for training style models. An important assumption here is that the paraphrase is stripped of its original style and does not leak into the training. The paper addresses this potential issue by training classifiers to predict style on both the original and paraphrased datasets and reporting the accuracy of trained models. We think this approach is most promising since it checks two important boxes - the system needs to be able to rewrite the text, the system also needs to be able to work on new styles without having to first collect a parallel dataset.

Application

We have been developing a web application which is a rich text editor that will provide stylistic analysis and prompts on text as users write. We have built an API in Flask to integrate our machine learning models with our web application. The models can run on a server and interface with the front end of our application through this API that we are developing.

Next Steps

There are several next steps that we are planning to take. These include:

- Additional Styles:

  We want to consider more aspects of style than the ones that we have experimented with so far (politeness, humor, and positive/negative sentiment) and we would like to be able to provide domain-specific style suggestions to users for specific domains of writing. We are considering Wikipedia (likely divided into broad topics) and Project Gutenberg as possible sources for domain specific corpora.

- Optimize Model Performance:

  We plan to improve the performance of our classifier models. If these classifier models are performing better, we can provide a more useful and accurate style analysis to users. We will do this by tuning hyperparamters of our classifier to increase their performance on the dataset we are using for validation using standard evaluation metrics of accuracy and F_1 score. In addition to increasing performance on standard benchmarks, we may also try to understand the model's performance on text samples that we would like or expect it to correctly classify and analyze. Such text samples would be annotated by us or other test users before the model sees them. This seems like a reasonable step, because we want our models to perform well not just on the distributions of text on which they were trained, but also on the distributions of text that users will provide it. Such distributions are very likely to be different and we should try to understand any distribution shifts in model performance that may occur.

- Style Transfer:

We will finish implementing an automated style transfer method and integrate it into the API and application. This model may be based on the tag and generate approach or the paraphrase back-mapping approach. When we generate the text in new styles, we would like to be able to provide multiple suggestions to the user so that they may select from a small set of options. This is keeping in line with the philosophy that we want our tool to foster interactive collaboration between human users and algorithmic suggestions, rather than just providing fully automated pipelines.

- Stylistic Auto-complete:

  We may still want to include a stylistic auto-complete feature that could suggest to users the next few words, given what they have written so far and their desired style(s). Based on the insights from our work so far, we think this feature may not be as useful for our application as the other aspects that we want to focus on. Therefore, we will prioritize this addition less than the other steps, but it still represents one possible evolution of our project that we may pursue.

- Empirical Analysis of Cross-Style Correlations and Dependence:

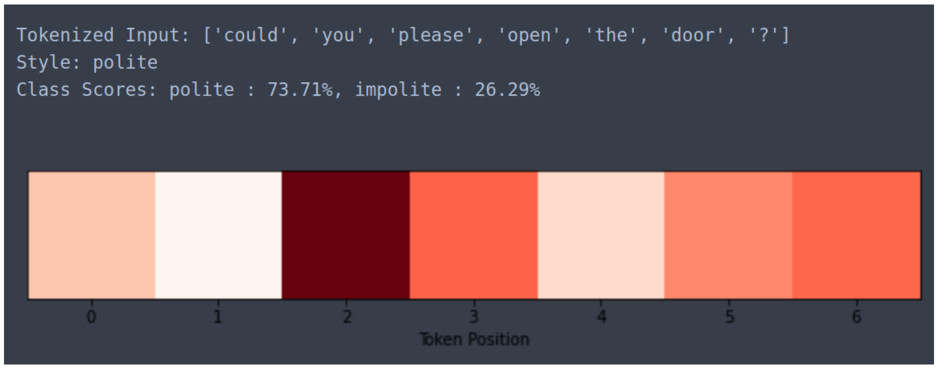
  As mentioned earlier, we would like to perform analysis similar to that done in xSLUE Kang and Hovy 2019 to understand how the different styles that we are considering are correlated with one another and the extent to which they have dependencies. We want to extend their analysis to thee style transfer and style editing that our application will perform.

- UI Enhancements:

  We have already integrated the heatmap of the attention visualization into the user interface of our application, but we still need to integrate the other features that we are developing and refine how we are displaying the style classification information. We also need to understand how use-able and useful the application is to users once it is more refined and make any adjustments as needed.

- Style Tuner: "Style Tuner" is one of the ambitious features we are aiming for our application. This will allow the user to design their own styles by using combinations of different styles. They can decide the weights of each style and get a coherent sentence based on their custom-designed styles.

Figures and Screenshots



```
Tokenized Input: ['could', 'you', 'please', 'open', 'the', 'door', '?']
Style: polite
Class Scores: polite : 73.71%, impolite : 26.29%
```

Figure 1: This is a visualization created while we were developing the heatmap method for visualizing the model's attention and explaining to users what is contributing to the predicted style. Here we were testing with a test sentence not from the dataset which we expected to be polite, "Could you please open the door?".

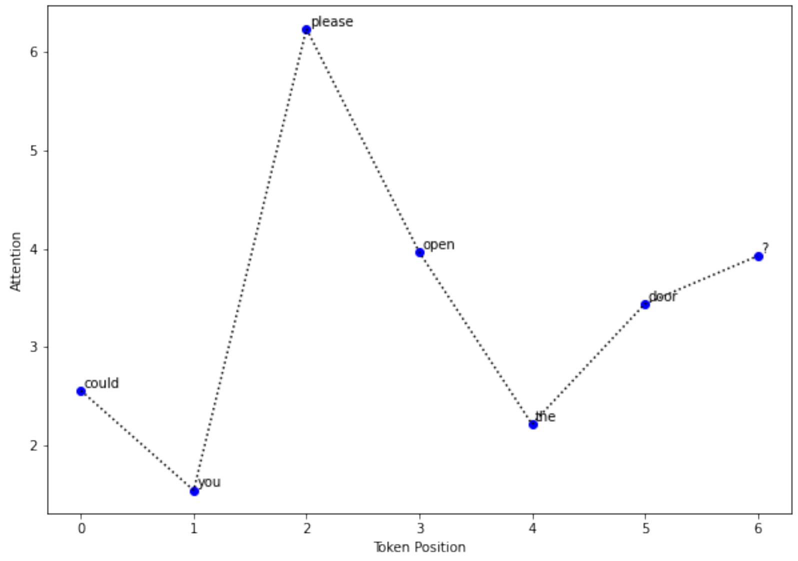
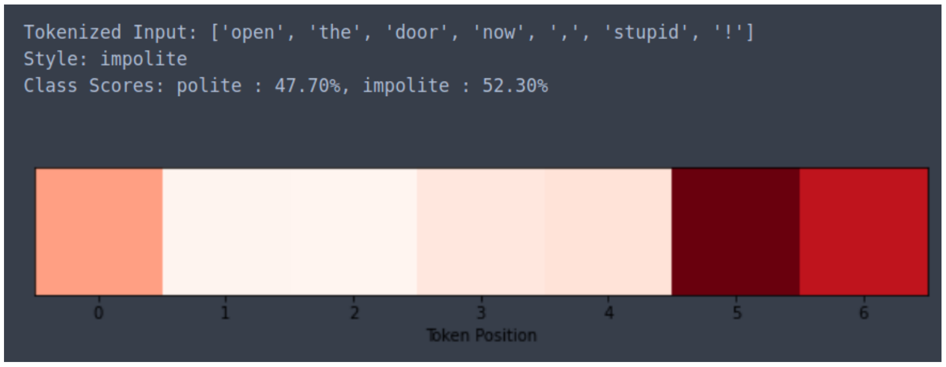


Figure 2: Plot of attention summed over 6 DistilBERT attention heads for predicting the politeness of the sentence ”Could you please open the door?”.



```
Tokenized Input: ['open', 'the', 'door', 'now', ',', 'stupid', '!']
Style: impolite
Class Scores: polite : 47.70%, impolite : 52.30%
```

Figure 3: This is a visualization created while we were developing the heatmap method for visualizing the model’s attention and explaining to users what is contributing to the predicted style. Here we were testing with a test sentence not from the dataset which we expected to be impolite, ”Open the door now, stupid!”

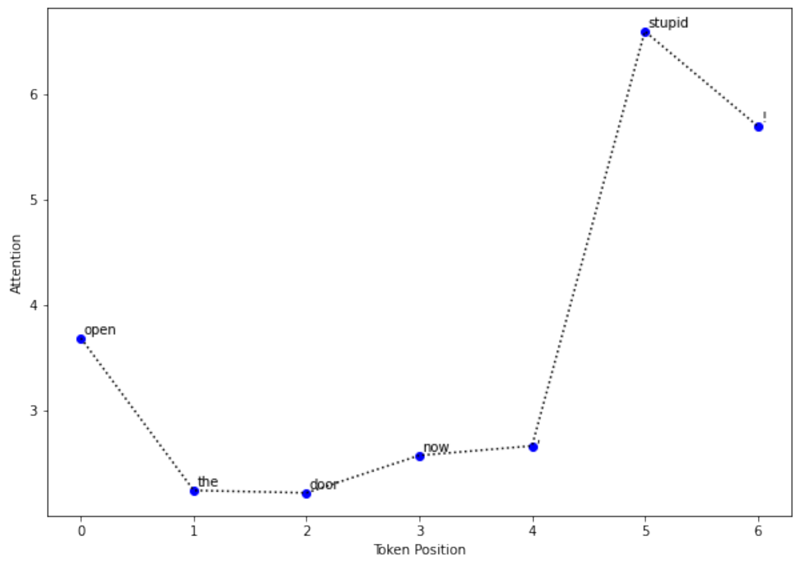


Figure 4: Plot of attention summed over 6 DistilBERT attention heads for predicting the politeness of the sentence ”Open the door now, stupid!”.

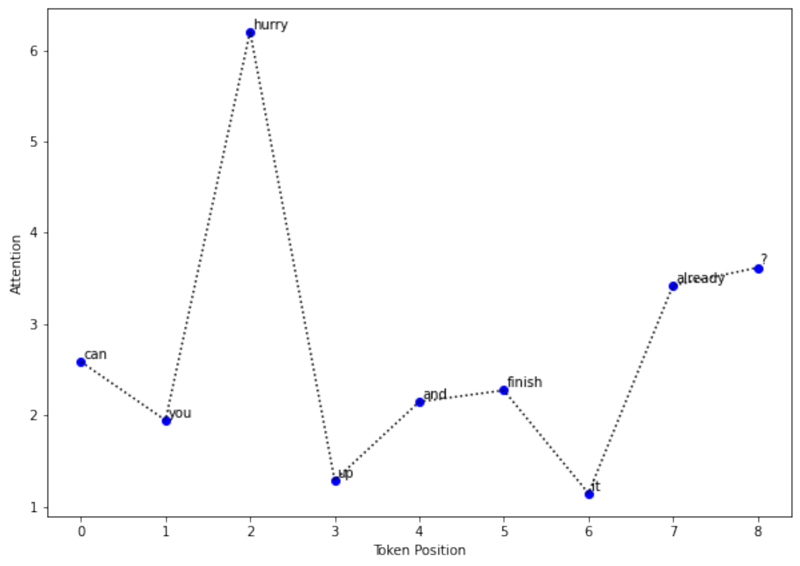


Figure 5: Plot of attention summed over 6 heads for predicting the politeness of the impolite sentence ”Can you hurry up and finish it already?”.

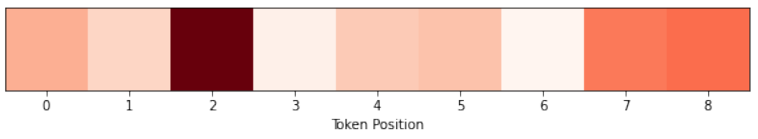


Figure 6: Heatmap of attention summed over 6 heads for predicting the politeness of the impolite sentence ”Can you hurry up and finish it already?”.

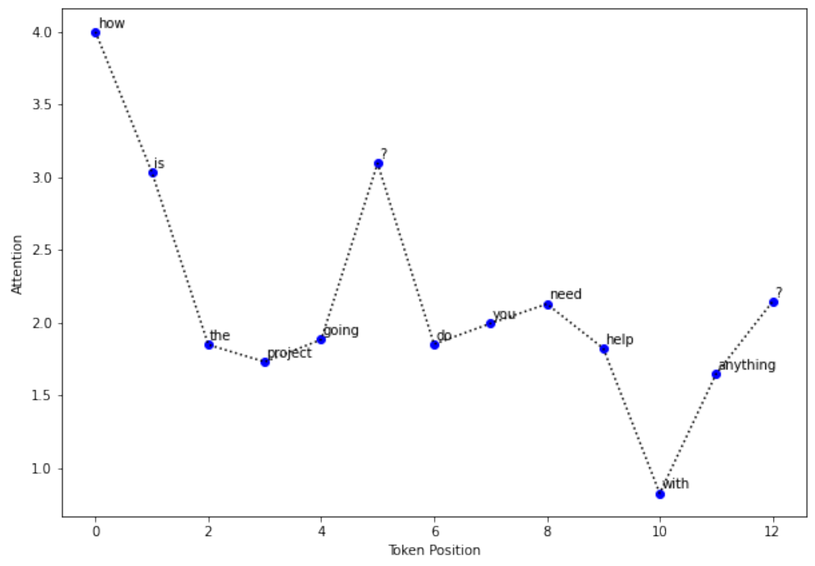


Figure 7: Plot of attention summed over 6 heads for predicting the politeness of the polite sentence "How is the project going? Do you need help with anything?".

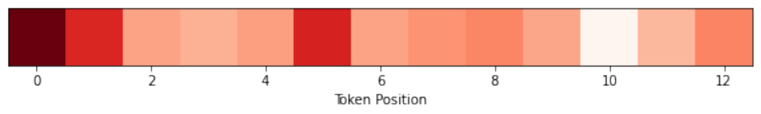
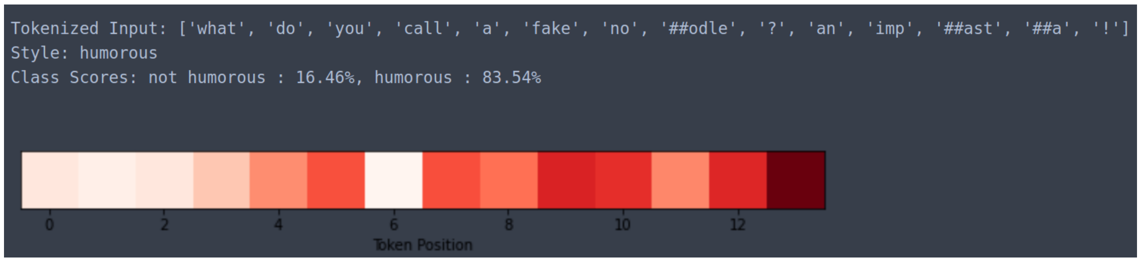


Figure 8: Plot of attention summed over 6 heads for predicting the politeness of the sentence "How is the project going? Do you need help with anything?".



```
Tokenized Input: ['what', 'do', 'you', 'call', 'a', 'fake', 'no', '##odle', '?', 'an', 'imp', '##ast', '##a', '!']
Style: humorous
Class Scores: not humorous : 16.46%, humorous : 83.54%
```

Figure 9: This is a visualization created while we were developing the heatmap method for visualizing the model's attention and explaining to users what is contributing to the predicted style. Here we were testing with a test sentence not from the dataset which we expected to be humorous, "What do you call a fake noodle? An impasta!"

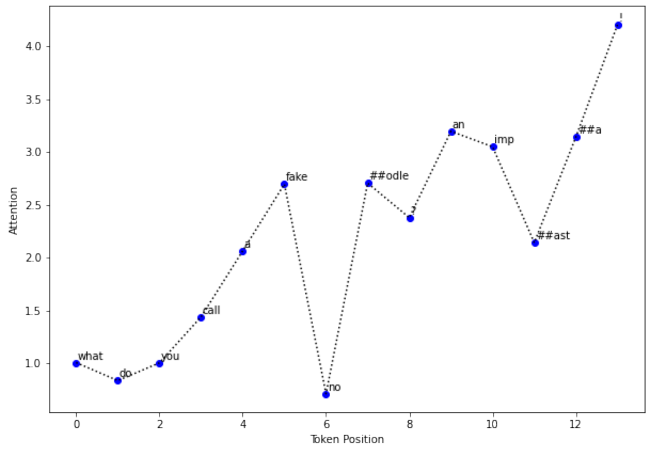
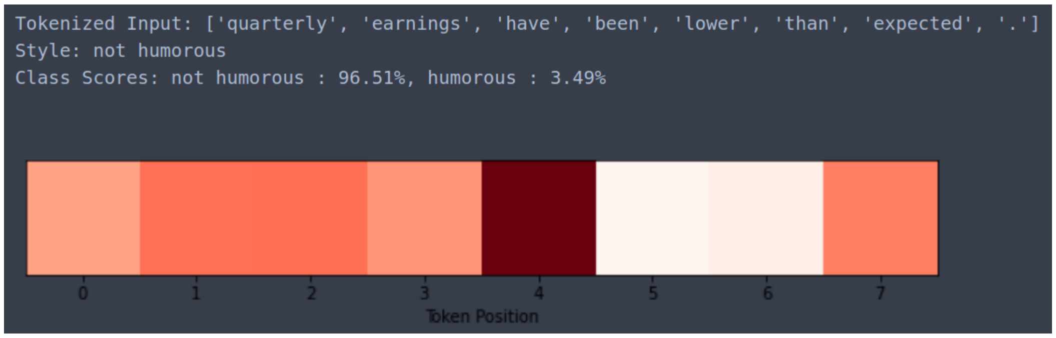


Figure 10: Plot of attention summed over 6 DistilBERT attention heads for predicting the politeness of the sentence "What do you call a fake noodle? An impasta!"



```
Tokenized Input: ['quarterly', 'earnings', 'have', 'been', 'lower', 'than', 'expected', '.']
Style: not humorous
Class Scores: not humorous : 96.51%, humorous : 3.49%
```

Figure 11: This is a visualization created while we were developing the heatmap method for visualizing the model's attention and explaining to users what is contributing to the predicted style. Here we were testing with a test sentence not from the dataset which we expected to be not humorous, "Quarterly earnings have been lower than expected."

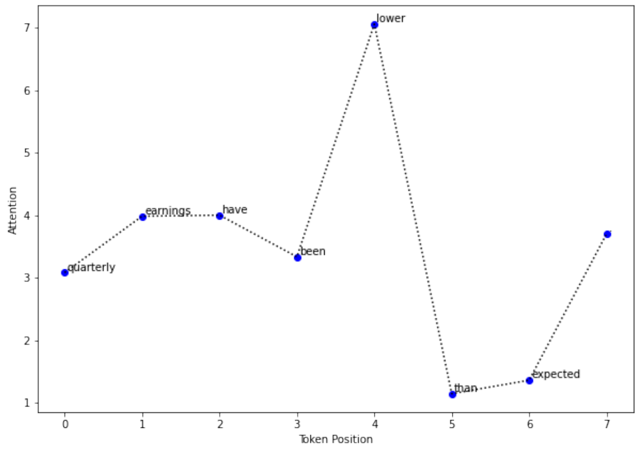


Figure 12: Plot of attention summed over 6 DistilBERT attention heads for predicting the politeness of the sentence "What do you call a fake noodle? An impasta!"

References

[BH81] M. Bakhtin and M. Holquist. *The dialogic imagination: Four essays.* Austin: University of Texas Press., 1981.

[Dan+13] Cristian Danescu-Niculescu-Mizil et al. "A computational approach to politeness with application to social factors". In: *Proceedings of the 51st Annual Meeting of the Association for Computational Linguistics (Volume 1: Long Papers).* Sofia, Bulgaria: Association for Computational Linguistics, Aug. 2013, pp. 250–259. URL: `https://www.aclweb.org/anthology/P13-1025`.

[Dat+20] Sumanth Dathathri et al. "Plug and Play Language Models: A Simple Approach to Controlled Text Generation". In: *International Conference on Learning Representations.* 2020. URL: `https://openreview.net/forum?id=H1edEyBKDS`.

[Dev+18] Jacob Devlin et al. "BERT: Pre-training of Deep Bidirectional Transformers for Language Understanding". In: *CoRR* abs/1810.04805 (2018). arXiv: `1810.04805`. URL: `http://arxiv.org/abs/1810.04805`.

[GEB15] Leon A. Gatys, Alexander S. Ecker, and Matthias Bethge. "A Neural Algorithm of Artistic Style". In: *CoRR* abs/1508.06576 (2015). arXiv: `1508.06576`. URL: `http://arxiv.org/abs/1508.06576`.

[KH19] Dongyeop Kang and Eduard H. Hovy. "xSLUE: A Benchmark and Analysis Platform for Cross-Style Language Understanding and Evaluation". In: *CoRR* abs/1911.03663 (2019). arXiv: `1911.03663`. URL: `http://arxiv.org/abs/1911.03663`.

[KWI20] Kalpesh Krishna, John Wieting, and Mohit Iyyer. "Reformulating Unsupervised Style Transfer as Paraphrase Generation". In: *Empirical Methods in Natural Language Processing.* 2020.

[Lin+17] Zhouhan Lin et al. "A Structured Self-attentive Sentence Embedding". In: *CoRR* abs/1703.03130 (2017). arXiv: `1703.03130`. URL: `http://arxiv.org/abs/1703.03130`.

[Mad+20] Aman Madaan et al. *Politeness Transfer: A Tag and Generate Approach.* 2020. arXiv: `2004.14257 [cs.CL]`.

[Pry+20] Reid Pryzant et al. "Automatically Neutralizing Subjective Bias in Text". In: *Proceedings of the AAAI Conference on Artificial Intelligence* 34.01 (Apr. 2020), pp. 480–489. DOI: 10.1609/aaai.v34i01.5385. URL: https://ojs.aaai.org/index.php/AAAI/article/view/5385.

[PSM14] Jeffrey Pennington, Richard Socher, and Christopher Manning. "GloVe: Global Vectors for Word Representation". In: *Proceedings of the 2014 Conference on Empirical Methods in Natural Language Processing (EMNLP)*. Doha, Qatar: Association for Computational Linguistics, Oct. 2014, pp. 1532–1543. DOI: 10.3115/v1/D14-1162. URL: https://www.aclweb.org/anthology/D14-1162.

[Ril+20] Parker Riley et al. *TextSETTR: Label-Free Text Style Extraction and Tunable Targeted Restyling*. 2020. arXiv: 2010.03802 [cs.CL].

[San+19] Victor Sanh et al. "DistilBERT, a distilled version of BERT: smaller, faster, cheaper and lighter". In: *CoRR* abs/1910.01108 (2019). arXiv: 1910.01108. URL: http://arxiv.org/abs/1910.01108.

[She+17] Tianxiao Shen et al. *Style Transfer from Non-Parallel Text by Cross-Alignment*. 2017. arXiv: 1705.09655 [cs.CL].

[Soc+13] Richard Socher et al. "Recursive Deep Models for Semantic Compositionality Over a Sentiment Treebank". In: *Proceedings of the 2013 Conference on Empirical Methods in Natural Language Processing*. Seattle, Washington, USA: Association for Computational Linguistics, Oct. 2013, pp. 1631–1642. URL: https://www.aclweb.org/anthology/D13-1170.

[Vas+17] Ashish Vaswani et al. "Attention Is All You Need". In: *CoRR* abs/1706.03762 (2017). arXiv: 1706.03762. URL: http://arxiv.org/abs/1706.03762.

[Vig19] Jesse Vig. "A Multiscale Visualization of Attention in the Transformer Model". In: *Proceedings of the 57th Annual Meeting of the Association for Computational Linguistics: System Demonstrations*. Florence, Italy: Association for Computational Linguistics, July 2019, pp. 37–42. DOI: 10.18653/v1/P19-3007. URL: https://www.aclweb.org/anthology/P19-3007.

[Wol+20] Thomas Wolf et al. “Transformers: State-of-the-Art Natural Language Processing”. In: *Proceedings of the 2020 Conference on Empirical Methods in Natural Language Processing: System Demonstrations*. Online: Association for Computational Linguistics, Oct. 2020, pp. 38–45. URL: `https://www.aclweb.org/anthology/2020.emnlp-demos.6`.