Understanding the Use of Inflammatory Headlines to Alter the Perception of an Article

Sami Rafat Amer (samiamer@mit.edu)
Junior, Course 6-9, EECS and BCS
All code can be found at: `https://github.com/sami-amer/headline-effects`

Abstract

In this project, I used *New York Times* article and headline data from 2003 to draw conclusions about the objectivity and tone of headline/article pairs. Through various methods in Python and WebPPL, I show that there is a clear difference in the objectivity of headlines and articles: headlines are consistently less objective, and lean more towards an emotive tone (positive or negative). The larger the difference in objectivity, the larger the difference in tonality between the headline and article. This trend held across the entire data set, with some outliers appearing when the difference in objectivity was small enough as to be considered insignificant. I also used a general mixture model to cluster the data, to confirm these same trends. I also used this mixture model with rejection sampling to control for specific variables, showing that our trends held even when we set one of the variables to some constant. While there were limitations to what I could feasibly analyze and compute, the data and analysis in this paper suggests that, across all *New York Times* articles from 2003, the headlines were consistently and significantly less objective and exhibited more tone than the article itself.

Keywords: NLP; GLoVe; New York Times; Sentiment Analysis; NLTK; SentiWord Net; Python; WebPPL; General Mixture Model

Introduction

One can argue that the headline is the most important part of the news article: it is likely to be read in full, even by those who do not intend to read the article, and it sets the lens through which the rest of the article is viewed. Knowing the power of the headline, news organizations and journalists curate them carefully and deliberately, oftentimes updating the headline as new information comes to light, sometimes vastly changing the perception of an article. With this in mind, I set out to find a trend between the objectivity and tonality of article/headline pairs. Through various methods of analysis, I hoped to find a concrete trend relating articles and headlines. I expected more emotive, inflammatory headlines to correspond to a larger gap in objectivity between the headline and the article's text, with more 'accurate', or appropriate, headlines corresponding to a smaller gap. In this project, I use a combination of methods from 9.66 as well as additional machine learning tools to analyze how headlines play with a reader's expectations, and how this often differs from the reliably more neutral reality presented in the article itself.

My analyses show, When the headline and article share sentiment, the author gets what they expect: an objective headline corresponds to an objective article. Otherwise a reader's expectation will be very different from the reality of the text. Understanding this is especially critical in today's click-bait culture, where readers often only read the headline while scrolling by, then disseminating this headline to others. The objectivity and tone of the headline are oftentimes more critical to the public perception of an event than the content of the article itself, and any purposeful alterations in the headline to make them more inflammatory can be seen as purposeful misrepresentation of the news.

Data set

The data set used was a month sub-set of the Concretely-annotated *New York Times* (Ferraro et al., 2014). I chose the month of March in the year 2003. The data set was pre-processed using Concrete, an NLP toolkit. As I did not need any of the Concrete-specific methods, I stripped all the Concrete annotation and used the data as plain-text.

Methods

To find a pattern between the sentiments of headlines and articles, I first needed to calculate some sort of sentiment score. In addition to analyzing the sentiment scores through visualization, I also wanted to see if there were any significant clusters within the data. I employed a variety of methods: sentiment scores were calculated and plotted with Python, while clustering relied on WebPPL (Goodman & Stuhlmüller, 2014).

Cleaning and Pre-processing the Text

Before any calculations were done, the text was cleaned, removing any non-alphanumeric characters, setting all letters to lowercase, and stripping any line breaks, indents, punctuation, or HTML characters. Next, the words were tokenized with NLTK's tokenizer, after which I removed all stop words that occured in the NLTK stop word dictionary. Finally, I lemmatized the tokens using the Word Net Lemmatizer from NLTK (Bird, Klein, & Loper, 2009).

Computing Sentiment Score

For sentiment analysis I set out to use the popular SentiwordNet (Esuli & Sebastiani, 2007), which was readily accessible in the Python NLTK package (Bird et al., 2009). SentiWordNet computes a positive, negative, and objective score per word, and does not take into account the overall sentence

structure. This was by design and not a short-coming: while there are tools that give the sentiment score of the entire text at once,I wanted to reduce the biases from pre-trained tools as much as possible. Additionally, I wanted the granularity of being able to work word for word. To do this over the course of a text, I iterated through each word, tallying each score (positive, negative, objective) and then averaging it over the number of words in the text. This gives the overall positivity, negativity, and objectivity of a piece of text. Objectivty and positivity ranged from 0 to 1, with closer to 1 meaning the text was more objective or more positive, respectively. Negativity ranged from 0 to -1, with a more negative score corresponding to a more negative article. Note that an article has all three scores at once, and due to the method of calculation a negative article still has a positivity score, and vice versa.

Plotting Sentiment Scores

To plot the data, I used Python packages MatPlotLib and Numpy (Hunter, 2007). To accurately plot trends, I used our previous scores to compute some new metrics:

1. **Objectivity ratio**, the ratio of the article's objectivity to the headline's objectivity (a higher score here meant the article was more objective)
2. **Objectivity difference**, which is the absolute value difference between objectivity of the article and headline
3. **Tonal difference**, which is the absolute value difference of the positive scores added to the absolute value difference of the negative scores.

Clustering

For clustering, I built off the general mixture model (Kemp, Loorbach, & Rotmans, 2007) from 9.66's fourth problem set. To maintain as much continuity between the problem set model and my new model, I ran all the code inside the problem set itself, which is browser-based. With some difficulty, I was able to force Safari (my browser of choice) to override the JavaScript code for visualization with my modified version of the code that added a wider color spectrum (expanded from 9 colors to 100), which allowed me to color-map the data directly in WebPPL. There was also an issue with importing data (see Limitations), and as such I was limited to 1250 samples, which I chose randomly from the 6000 samples in the data set. I plotted this data to confirm that the distribution was representative of the total data set and then used Python to create a list of dictionaries, each of which had tonal difference, headline objectivity, objectivity difference, and a color calculated based on the objectivity difference. I then turned this list of dictionaries to a string, and copy-pasted the plain-text into WebPPL. Due to the similarities between JavaScript objects and Python dictionaries, I was able to assign this to a variable directly.

While the general underlying model of the problem set was similar to what I wanted, I had to change a significant amount of the code to account for the difference in date (mostly a jump from categorical distributions to Gaussian distributions). I changed all of the categorical distributions to Gaussian, with mu and sigma derived from the set of 1250 samples. I also changed the number of categories to 20, as I expected more classes due to the large amount of data.

After changing the code, I ran a Markov-Chain Monte Carlo inference, and plotted the data.

Looking for Bias

To look for a trend in headline-article bias, I again borrowed methodology from problem set four, this time building my code off of the `ImagineColor` code. The idea was to imagine away, or control, each of the three variables in my data set, and see if the model had any inherent bias. Additionally, the plots could tell us more about the the relationship between these variables.

To achieve this, I created three functions: `ImagineObjDiff`, `ImagineHeadlineObj`, and `ImagineTonalDiff`. Similarly to the problem set, I used rejection sampling to plot more efficiently.

Analysis

Analysis of the Data in Python

I first created a scatter plot which mapped the tonal difference to the objectivity difference, which can be seen in Figure 1.

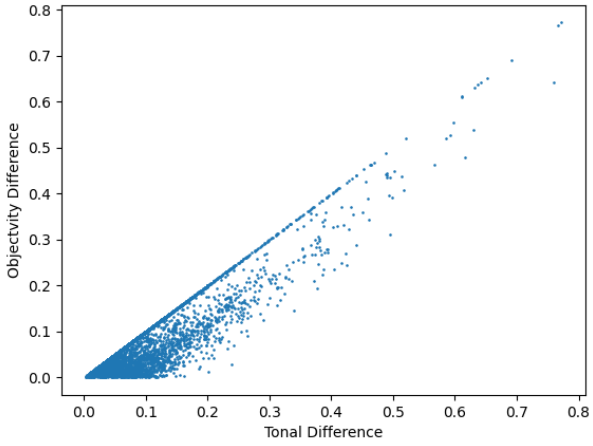

Figure 1: Scatter plot of the Tonal Difference vs Objective Difference

Here we see a large clustering towards the bottom-left of the page due to a large amount of articles that had a smaller objectivity difference; this specific subset of noise becomes more apparent in the subsequent quiver plots, as well as the WebPPL plots (shown later in this section). In this scatter plot, however, we can clearly see tonal difference increasing linearly with objectivity difference. This means as the delta in tone between an article and its headline increases, the delta in objectivity between the article and its headline increases

as well. This plot clearly shows that less objective headlines tend to have less objective tones.

Wanting to pursue this specific trend further, I created the quiver plot you see in Figure 2. This quiver plot was created manually using MatPlotLib lines and arrows: this was a deliberate choice to properly contextualize the magnitudes of the arrows.

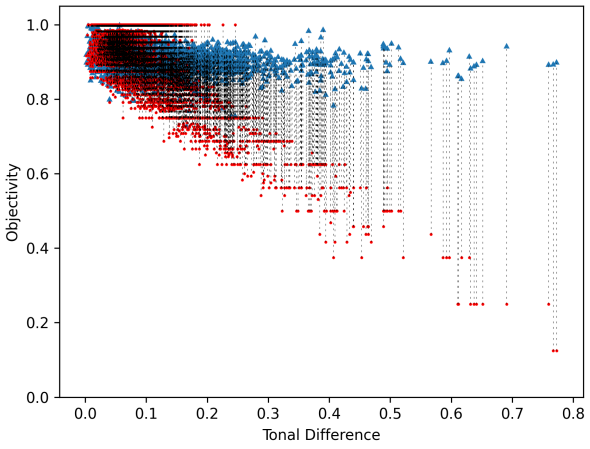

Figure 2: Quiver plot of the tonal difference and objectivity

In this quiver plot, I mapped the arrows from the headline objectivity (red) to the article objectivity (blue) in an attempt to discern the magnitude of difference between the tone and objectivity of each headline/article pair. The large majority of the arrows point up, representing the increase in objectivity between the headlines and the articles (i.e. articles are usually more objective than their headlines). There were some arrows pointing down, but these were limited to the subset of noise in the top-left, where the objectivity of the article and headline are very close. So, these cases were of such small size as to be insignificant and related to small uncertainties in the objectivity and tone calculations.

Notice also the magnitude of the arrows: when the tonal difference is large, so is the difference in objectivity, with the headline being significantly less objective. This may seem straightforward, but remember that we calculated these scores in such a way that a piece of text had all 3 scores (positive, negative, objective) independent of each other. This shows adding an "emotive" or otherwise inflammatory headline creates a very large gap between the expectation and reality of the article.

Having established that there is seemingly some trend in the difference in objectivity between headlines and articles, I began plotting the positive and negative scores to see if the tone had similar trends. To do this, I created quiver plot that plotted an arrow for every article based on its positivity and negativity score. The base of the arrow represents the headline's tone and the tip represents the article's tone. I also color mapped the plot based on the arrow size, with a color bar on the right that maps color to arrow length. This can all be seen in Figure 3. This quiver plot was made using MatPlotLib's quiver function, and as such the arrows are not to scale, but are still scaled relatively, so that one arrow may be compared to the next.

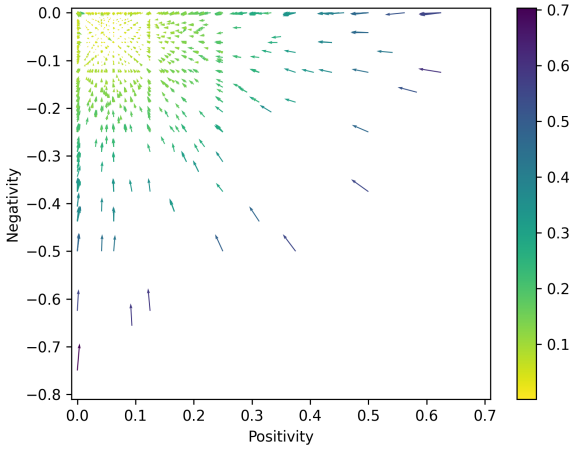

Figure 3: Quiver plot of the positive and negative score, color-mapped to arrow size

Here again we see a consistent arrow direction, this time towards the top-left, which signifies the articles are often less negative and less positive than their headlines. No matter where the headline falls on the scale, the articles consistently show less tone. We can also see the magnitude of change get larger the further away from neutral tone we get: headlines that are very positive, very negative, or a lot of both are much further from the scores of their respective articles than headlines that are closer to neutral in tone.

For another view on this same plot, I changed the color mapping to represent the objectivity ratio instead of the arrow size. To more clearly define the differences, I log-normalized the color mapping. All of this can be seen in Figure 4

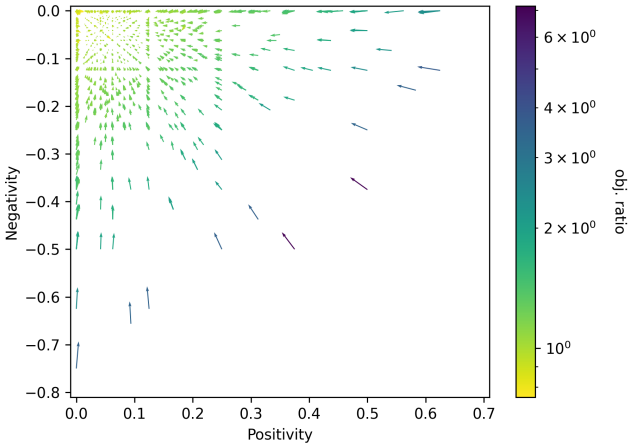

Figure 4: Quiver plot of the positive and negative score, color-mapped to obj. ratio

With the log-normalized colormap, we can see the trend in objectivity difference matches almost exactly the trend in tone. As we might expect, the larger arrows had a larger objectivity ratio, which aligns with our expectations of the article being more objective than the headline. We can see articles with highly positive or highly negative headlines also had larger objectivity differences between the article and headline.

Analysis with WebPPL

I now applied the clustering analysis in WebPPL to validate the trends in objectivity difference and tonal difference I was seeing visually in my plots. To begin the WebPPL analysis, I first plotted the data based on its headline objectivity vs tonal difference, using a colormap based on the objectivity difference imported from Python. This data is a randomly selected sub-sample of 1250 article/headline pairs. The color represents the objective difference, with red being smallest and violet being largest (I used the `gist_rainbow` color map). We can see this plot in Figure 5.

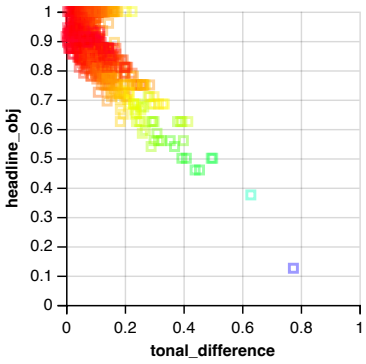

Figure 5: Our color mapped data as plotted in WebPPL

This plot is directly analogous to our quiver plots in Python and validates the data was correctly imported into WebPPL. I then ran the general mixture model with $K = 20$, with Gaussian distributions for the headline objectivity, tonal difference, and objectivity difference. I computed μ and σ for the Gaussian distributions based on the mean and standard deviation of my article data in Python. The general mixture model created fifteen classes, which can be seen below. A sample of six classes is shown in Figure 6 below.

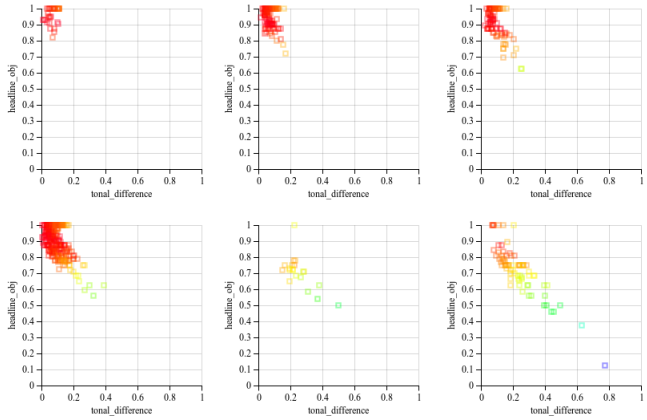

Figure 6: Six sample (out of fifteen) clusters identified using a general mixture model in WebPPL.

I initially expected the data to cluster based on objectivity difference, and that we would see clear clusters according to the color. While we see some interesting clusters and they definitely show a strong trend with color (i.e. objectivity difference), we find that the predisposition of the data to cluster at the top-left heavily skews what clusters the model can find. I attempted to alleviate this by varying the μ and σ values, but to no avail. However, while this subset of noise affected the clustering, it mostly made redundant clusters. What unique clusters we do have match how we would initially expect the data cluster.

For further analysis, I controlled for one variable at a time (by setting it to some constant) and used rejection sampling to plot the remaining two variables. I first did this for objective difference, setting it to 0.3 and generating 100 samples, as can be seen in Figure 7.

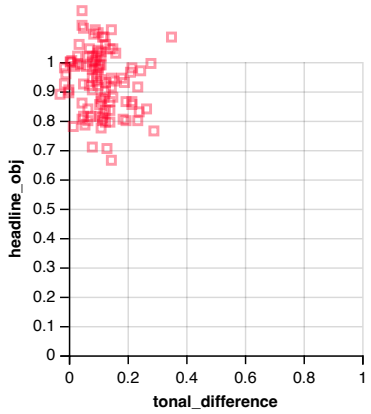

Figure 7: ”Imagining” all objective differences to be the same

For a low objective difference (like our fixed parameter 0.3), we expect an objective headline and a low tonal difference between article and headline. Looking at the plot, we can see that it matches this expectation, with a cluster located at the top left of the graph (remember that we controlled for objective difference, which is represented through color. As such, all of the data points are the same color).

I controlled for tonal difference by setting it to 0.2, generating Figure 8. The plot again confirms our expectations from earlier analysis, showing that at a low tonal difference we have higher headline objectivity. We can also notice that the data defines some range of headline objectivity from 0.7 onward. I suspect that this would hold with more samples, but was not able to test this (see Limitations). Notice also the color (which corresponds to objectivity difference): while it varies, it does not stray too far from red (low objectivity difference). This accurately reflects our earlier quiver plots, which had the smaller arrows in the top left.

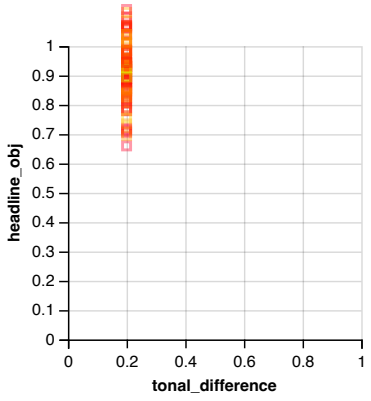

Figure 8: ”Imagining” all tonal difference to be the same

Finally, I controlled for headline objectivity, setting it to 0.75. Initially, I got the cluster we see in Figure 9. To further validate this tight cluster, I ran the rejection sampling again for 100 samples, which is what we see in Figure 10. While the cluster gets a little wider, it remains significantly tight, showing that at a high headline objectivity, we can expect less tonal difference in the article/headline pair. Additionally, we can see that the colors cover a range similar to Figure 8, again representing low objectivity difference.

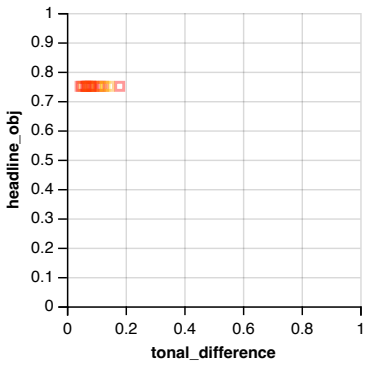

Figure 9: ”Imagining” all headline objectivity to be the same, with 20 samples

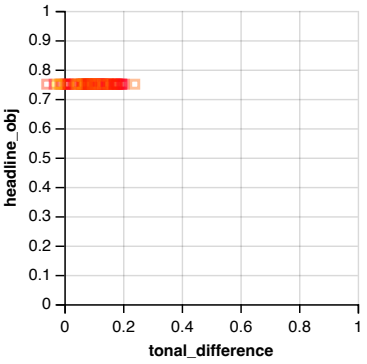

Figure 10: ”Imagining” all headline objectivity to be the same, with 100 samples

Conclusions

Through multiple methods of data generation and analysis, I showed that there is a clear difference in the objectivity of headlines and articles: headlines are consistently less objective, and lean more towards a tone (positive or negative). The larger the difference in objectivity, the larger the difference in tonality between the headline and article. This trend held across the entire data set, with some outliers appearing when the difference in objectivity was small enough as to be considered insignificant. This trend is significant; it shows that the *New York Times* headlines in our data set, whether purposefully or accidentally, are much more inflammatory than the content in their respective articles. This is problematic as it significantly alters how an article is read, and even more so how a news event is viewed. The gulf in objectivity leads to a

serious misrepresentation of an article or event, especially in today's click-bait culture that does not reward thorough reading or fact-checking of articles, instead focusing on bite-sized news. As we move towards quicker and quicker methods for consuming media, it is critical we keep in mind the effects of headlines, maintaining their objectivity within the scope of the related text to paint an accurate picture to both the thorough reader and the passer-by

Limitations and Future Work

Data and Computation Limitations I encountered many limitations throughout the course of this project, which if fixed could lead to stronger results. The most apparent of these limitations is the choice of data: I chose a narrower set of one-month to keep the computational cost reasonable. With no GPU-acceleration or multi-core CPU, anything more would have consumed too much of the project time-line. In addition to the data being computationally-expensive to generate, the move from Python to WebPPL forced me to select a sub-sample of 1250; this was due to the browser-based nature of the environment, which ran into a maximum recursion issue with anything more. Setting up a larger environment would allow us to use more of our data, allowing us to make stronger conclusions. In addition to the data limitation in WebPPL, I ran into significant resource-related issues, which were most apparent when controlling for specific variables in our rejection sampling. Attempting to run rejection sampling for ranges of data that were more sparse in our original data set froze the program and spiked the CPU past its limit. Similar to the data generation, more computational power here would allow us to run more analysis and reach stronger conclusions.

Data Noise As can be seen throughout the paper, the data was very noisy – as most articles and headlines had little objectivity difference – and skewed heavily to smaller articles with little objective or tonal difference. I believe these to be smaller, less significant – although common – articles, like obituaries or updates. Devising some method to sample more equally across the distribution would reduce this noise, allowing us to more easily analyze our data without seeing it skew towards this sub-set of noise.

Semantic Similarity Initially, this work used semantic similarity between articles and headlines. I calculated this using PyTorch and GLoVe embeddings, generating some metric for the similarity of the article/headline pairs. This metric proved to be too noisy, and was of no significance when we plotted it against other variables. In future work, I would hope to devise a better way to calculate this semantic similarity, as I believe there is important analysis to be done regarding semantic similarity and objectivity.

Acknowledgments

I would like to thank my girlfriend Holly Jackson for her help throughout the project and inspiration from some of her previous work (Jackson, 2021; Holly M. Jackson, 2021) and Professor Josh Tenenbaum for guidance throughout the semester.

References

Bird, S., Klein, E., & Loper, E. (2009). *Natural language processing with python: analyzing text with the natural language toolkit*. O'Reilly Media, Inc.

Esuli, A., & Sebastiani, F. (2007). *SentiWordNet: A High-Coverage Lexical Resource for Opinion Mining* (Tech. Rep. No. 02). Istituto di Scienza e Tecnologie dell'Informazione.

Ferraro, F., Thomas, M., Gormley, M., Wolfe, T., Harman, C., & Durme, B. V. (2014). Concretely annotated corpora. In *NIPS Workshop on Automated Knowledge Base Construction (AKBC)*. Retrieved from https://doi.org/10.35111/xgs8-5140

Goodman, N. D., & Stuhlmüller, A. (2014). *The Design and Implementation of Probabilistic Programming Languages*. dippl.org. (Accessed: 2021-12-13)

Holly M. Jackson. (2021, May). New York Times *Content Analysis*. Retrieved from github.com/hollyjackson/NYT_Content_Analysis

Hunter, J. D. (2007). Matplotlib: A 2d graphics environment. *Computing in Science & Engineering*, *9*(3), 90–95. doi: 10.1109/MCSE.2007.55

Jackson, H. M. (2021, May). *The* New York Times *Distorts the Palestinian Struggle: A Case Study of Anti-Palestinian Bias in American News Coverage of the First and Second Palestinian Intifadas*. (Preprint at web.mit.edu/hjackson/www/The_NYT_Distorts_the_Palestinian_Struggle.pdf)

Kemp, R., Loorbach, D., & Rotmans, J. (2007). Transition management as a model for managing processes of co-evolution towards sustainable development. *International Journal of Sustainable Development & World Ecology*, *14*(1), 78-91. Retrieved from https://doi.org/10.1080/13504500709469709