Chloe Hutchins, Monica Kavthekar and Jennifer Wright
Senior Design Writing 3
Due: 10/22/2019

Proposal For Bias Detection Web Plugin

Introduction

Machine learning, although common in the backend for several technology-based systems, has yet to become widespread at the simple level of a web plugin. By creating a web based machine learning application which will allow users to detect bias through a simple, easy-to-use interface that is accessible to all through a web extension, our senior design project will help revolutionize the way the general public uses and perceives machine learning. Our project will consist of three parts: a web scraper, a machine learning algorithm, and a web plugin frontend. All three of these technologies, used together, will create a new product that allows bias detection to not only be available to companies and students, but to the general public as well.

Technical Feasibility and Innovation

The first aspect of our project involves gathering data from web corpora to be used in the bias detection algorithm. In order to do the web scraping aspect of our project we use python; specifically, we will use the Selenium and BeautifulSoup libraries. Selenium is an API web driver that allows users to open a browser and get data from a page given a URL. Selenium is a great tool to use for web scraping because it provides functionality and simplicity. BeautifulSoup pulls data from HTML web pages and is perfect for the use case of just getting the words to be processed in the algorithm. By creating a simple, python-based web scraper, our team is making data collection easy. Our machine learning algorithm will easily process text scraped from the internet.

The bias detection algorithm will work in a series of steps. First, the words will be passed into Stanford University's GloVe algorithm. This will represent words as vector word embeddings, which will then allow them to be compared to each other in a meaningful way. Then, we will use the WEAT and WEFAT algorithms to test if each word is related to a certain side of a stereotype. We will begin with the male or female association, and then we will choose a particular stereotype. To begin, we have decided to focus on career versus family. This comparison will allow us to determine if the sentence relates more to men being career focused or women being family focused. We will then use our statistics to determine if the sentences related to male are more closely related to career or family or neither, and if the sentences related to female are more closely related to career or family. From there, we can create a matrix with the results and highlight the sentences on the page that are highly biased. We will highlight the sentences that perpetuate stereotypes in red based on their degree of cosine similarity to the stereotype. Then, we will highlight the sentences that work against the predicted bias in green. The WEAT and WEFAT algorithms use the biases that are identified from the implicit association test. Using the test removes the need for our team to decide what is and what is not biased. After developing the bias detection algorithm for one known bias, we will begin to build out the rest of the bias detection one bias at a time.

Much of the existing work about bias in the web is simply a discussion about the impact biased machine learning could have on a tech-driven future. There are very few algorithms that exist to detect bias in mass texts, especially using known methods like the implicit association test. Further, very few methods exist that allow the general public to access bias detection in their everyday lives. Because most people read the news or other texts on the web multiple times a day, it is crucial that people become

aware of the biases that are subconsciously being passed to them through these texts. Accessibility is the core reason that our product will be innovative for the future of bias detection in machine learning.

One of the tools that exists today to help detect bias in algorithmic data is the FairML framework. The developers of FairML identified the same problems that our group did- machine learning inherits the bias of past data, which is inherently unfair to previously marginalized groups. However, unlike our team's web plugin, FairML tackles the problem at the algorithmic level. Our plugin will identify and visualize existing bias, giving the general public a better idea of the bias that exists. FairML is targeted at developers looking to eliminate machine learning bias from models that they develop for other purposes. FairML classifies the features of a machine learning model into categories, including a group for a "protected" category that consists of gender, race, and other typically stereotyped groups. Then, the framework looks at the machine learning algorithm and determines if a feature from the protected category carries too much significance in the model's results. From there, FairML determines if the model relies too heavily on the protected feature, which allows it to tell the user if the model is biased. The framework is very important in eliminating bias at the core, but it serves a different purpose than our team's web plugin. FairML is targeted towards developers and companies, but our web plugin is targeted towards students and the general public. Our plugin will utilize techniques like FairML to ensure that the machine learning of our backend is not biased, but our product itself presents a new and innovative idea.

After developing a back-end algorithm for our project, we will develop a front end web plugin to visualize bias on the screen. Web plugins in Chrome are developed using JavaScript, HTML and CSS. Since our team wanted to develop the web extension to be compatible with a python back-end, we found a way to compile our code to JavaScript. For this, we will be using the tool Rapydscript. Rapydscript will allow our code to be compatible with Chrome. We will also be using HTML and CSS to display a dialog box with statistics and information about the bias in the text. The back end of our algorithm will kick off the web scraping, which will be passed into the bias detection algorithm. The web plugin will complete the package, allowing users to easily visualize bias on the page.

Creating a simple tool to visualize bias is innovative because it will enable users of all levels to physically see the stereotypes in their text without having to read and analyze any text. Currently, there is a Chrome web extension called *Bias Finder* that detects political bias in the news. *Bias Finder* is especially important for the current American political climate, because the perpetuation of fake news and the two-party political divide reaches into several aspects of society. *Bias Finder* pulls general data about news sources and displays it to a user; for example, the extension will tell its users if a news website tends to lean more to the left or right. This functionality is similar to our plugin's intended purpose. It gives the general public an overview of the bias on a website. However, our plugin is more innovative because it applies to bias as a more general term, and it is narrowly focused on political bias. Our tool is also innovative because it utilizes machine learning. *Bias Finder* has a backend that simply analyzes headlines and statements and identifies if they lean to the left or right of the simple facts. In addition to identifying if a certain statement aligns with a certain stereotype, our bias detection web plugin will also tell the user if the stereotypes are positive or negative. Finally, utilizing machine learning will add innovation to our product, giving it a more well-rounded outlook on the bias in a text and not limiting it to political perspectives. Thus, while the existing web technology for bias detection is certainly important, our team's web plugin will bring new innovation to the market.

Costs, Risks, and Risk Mitigation

As described above, there are three main parts to this project: the web scraper, the machine learning algorithm, and the web plugin. The machine learning algorithm is the part of the project that is expected to make up most of the code. This is because the machine learning algorithm is the more complex part of the project and the part that will be executing most of the product's functionality. Our code for the machine learning algorithm is based on preexisting code, which has about 1,000 lines. With the increased functionality added to the project the estimated code length is around 2,000 lines. For the hardware development costs, because the project is software based, our project will not have any hardware development costs. For the software development costs, much of the software and data that is being used is readily available for free, or we will complete it, so there will be no monetary software costs. Instead, the software development costs will be the time spent developing and testing the product. Based on the average time that will be spent working on the project, which is around 10 hours a week, the estimated software cost of the project is about 700 hours.

The phases of this project are planning and research, development, integration, adding functionality, and finally testing. The planning and research phase of this project has already been completed. Based on this phase the functionality and different aspects of the product were determined, as well as that the plan for how will be developed was created. The phase of the project that we are currently on is the development phase, which should be completed by the end of October. For this phase, basic versions of the web scraper, machine learning algorithm, and web plugin are being developed separately. The end result of this phase should be to have a very simple working version of the web scraper, machine learning algorithm, and web plugin. The next phase of the project is going to be to integrate each the parts of the project that have been developed. This part should be completed by mid-October. Once the parts are integrated, more functionality can be added to the product. This phase should be completed by the end of January. Throughout this phase as functions are added they will be tested as they are added. Once all of the necessary functions are added to the product, the testing phase will begin. This testing will be much more rigorous than the testing that occurred during the previous phase. This phase should be done by the end of March and will be the last phase before the product is introduced to the public.

Conclusion

Using the technologies above, we are confident that we will be able to implement a bias detection web plugin that has both the visual and analytical components to deliver an easily digestible and useful summary of the bias in text on a webpage. Our product will innovate further than existing products, which exist primarily to help developers eliminate bias in their machine learning models or to identify political biases. Using three components: a web scraper, machine learning algorithm, and a web plugin to visualize bias, our team will be able to accomplish our goal.

Sources

https://dzone.com/articles/bias-detection-in-machine-learning-models-using-fa
https://chrome.google.com/webstore/detail/bias-finder/jojjlkfeofgcjeanbpghcapjcccbakop?hl=en-US