Capstone Project Proposal

April 14, 2019

1 Unintended Bias in toxicity classification

1.1 Udacity Machine Learning Engineer Nanodegree

- *Author: Giuseppe Romagnuolo*
- *Project from the Kaggle competition: Jigsaw unintended bias in toxicity classification*
- *Field: Natural Language Processing*

1.2 Domain Background

Natural Language Processing is a complex field which is hypothesised to be part of AI-complete set of problems, implying that the difficulty of these computational problems is equivalent to that of solving the central artificial intelligence problem making computers as intelligent as people.[*cit.Wikipedia*]

With over 90% of data ever generated being produced in the last 2 years[*ref.ScienceDaily*] and with a great proportion being human generated unstructured text there is an ever increasing need to advance the field of Natural Language Processing.

Recent UK Government proposal to have measures to regulate social media companies over harmful content, including "substantial" fines and the ability to block services that do not stick to the rules is an example of the regulamentory need to better manage the content that is being generated by users. [*ref.BBC*]

Other initiatives like Riot Games's work aimed to predict and reform toxic player behaviour during games[*ref.ArsTechnica*] is another example of this effort to understand the content being generated by users and moderate toxic content.

However, as highlighted by the Kaggle competition Jigsaw unintended bias in toxicity classification, existing models suffer from unintended bias where models might predict high likelihood of toxicity for content containing certain words (e.g. "gay") even when those comments were not actually toxic (such as "I am a gay woman"), leaving machine only classification models still sub-standard.

Having tools that are able to flag up toxic content without suffering from unintended bias is of paramount importance to preserve Internet's fairness and freedom of speech.

1.3 Problem Statement

From Kaggle competition page:

The Conversation AI team, a research initiative founded by Jigsaw and Google (both part of Alphabet), builds technology to protect voices in conversation. A main area of focus is machine learning models that can identify toxicity in online conversations, where toxicity is defined as anything rude, disrespectful or otherwise likely to make someone leave a discussion.

Last year, in the Toxic Comment Classification Challenge, participants built multi-headed models to recognize toxicity and several subtypes of toxicity. This year's competition is a related challenge: building toxicity models that operate fairly across a diverse range of conversations.

Here's the background: When the Conversation AI team first built toxicity models, they found that the models incorrectly learned to associate the names of frequently attacked identities with toxicity. Models predicted a high likelihood of toxicity for comments containing those identities (e.g. "gay"), even when those comments were not actually toxic (such as "I am a gay woman"). This happens because training data was pulled from available sources where unfortunately, certain identities are overwhelmingly referred to in offensive ways. Training a model from data with these imbalances risks simply mirroring those biases back to users.

In this competition, you're challenged to build a model that recognizes toxicity and minimizes this type of unintended bias with respect to mentions of identities. You'll be using a dataset labeled for identity mentions and optimizing a metric designed to measure unintended bias. Develop strategies to reduce unintended bias in machine learning models, and you'll help the Conversation AI team, and the entire industry, build models that work well for a wide range of conversations.

1.4 Datasets and Inputs

From Kaggle dataset description:

Disclaimer: The dataset for this competition contains text that may be considered profane, vulgar, or offensive.

1.4.1 Background

At the end of 2017 the Civil Comments platform shut down and chose make their ~2m public comments from their platform available in a lasting open archive so that researchers could understand and improve civility in online conversations for years to come. Jigsaw sponsored this effort and extended annotation of this data by human raters for various toxic conversational attributes.

In the data supplied for this competition, the text of the individual comment is found in the `comment_text` column. Each comment in Train has a toxicity label (`target`), and models should predict the `target` toxicity for the Test data. This attribute (and all others) are fractional values which represent the fraction of human raters who believed the attribute applied to the given comment. For evaluation, test set examples with `target >= 0.5` will be considered to be in the positive class (toxic).

The data also has several additional toxicity subtype attributes. Models do not need to predict these attributes for the competition, they are included as an additional avenue for research. Subtype attributes are:

- severe_toxicity
- obscene
- threat
- insult
- identity_attack
- sexual_explicit

Additionally, a subset of comments have been labelled with a variety of identity attributes, representing the identities that are *mentioned* in the comment. The columns corresponding to identity attributes are listed below. Only identities with more than 500 examples in the test set (combined public and private) will be included in the evaluation calculation. These identities are shown in bold.

- male
- female
- transgender
- other_gender
- heterosexual
- homosexual_gay_or_lesbian
- bisexual
- other_sexual_orientation
- christian
- jewish
- muslim
- hindu
- buddhist
- atheist
- other_religion
- black
- white
- asian
- latino
- other_race_or_ethnicity
- physical_disability
- intellectual_or_learning_disability
- psychiatric_or_mental_illness
- other_disability

Note that the data contains different comments that can have the exact same text. Different comments that have the same text may have been labeled with different targets or subgroups.

1.4.2 Examples

Here are a few examples of comments and their associated toxicity and identity labels. Label values range from 0.0 - 1.0 represented the fraction of raters who believed the label fit the comment.

Comment: *i'm a white woman in my late 60's and believe me, they are not too crazy about me either!!*

- Toxicity Labels: All 0.0
- Identity Mention Labels: female: 1.0, white: 1.0 (all others 0.0)

Comment: *Why would you assume that the nurses in this story were women?*

- Toxicity Labels: All 0.0
- Identity Mention Labels: female: 0.8 (all others 0.0)

Comment: *Continue to stand strong LGBT community. Yes, indeed, you'll overcome and you have.*

- Toxicity Labels: All 0.0
- Identity Mention Labels: homosexual_gay_or_lesbian: 0.8, bisexual: 0.6, transgender: 0.3 (all others 0.0)

In addition to the labels described above, the dataset also provides metadata from Jigsaw's annotation: `toxicity_annotator_count` and `identity_annotator_count`, and metadata from Civil Comments: `created_date`, `publication_id`, `parent_id`, `article_id`, `rating`, `funny`, `wow`, `sad`, `likes`, `disagree`. Civil Comments' label `rating` is the civility rating Civil Comments users gave the comment.

1.4.3 Labelling Schema

To obtain the toxicity labels, each comment was shown to up to 10 annotators. Annotators were asked to: "Rate the toxicity of this comment"

- Very Toxic (a very hateful, aggressive, or disrespectful comment that is very likely to make you leave a discussion or give up on sharing your perspective)
- Toxic (a rude, disrespectful, or unreasonable comment that is somewhat likely to make you leave a discussion or give up on sharing your perspective)
- Hard to Say
- Not Toxic

These ratings were then aggregated with the `target` value representing the fraction of annotations that annotations fell within the former two categories.

To collect the identity labels, annotators were asked to indicate all identities that were mentioned in the comment. An example question that was asked as part of this annotation effort was: "What genders are mentioned in the comment?"

- Male
- Female
- Transgender
- Other gender
- No gender mentioned

Again, these were aggregated into fractional values representing the fraction of raters who said the identity was mentioned in the comment.

The distributions of labels and subgroup between Train and Test can be assumed to be similar, but not exact.

1.4.4 File descriptions

- train.csv - the training set, which includes subgroups
- test.csv - the test set, which does not include subgroups
- sample_submission.csv - a sample submission file in the correct format

1.4.5 Usage

This dataset is released under CC0, as is the underlying comment text.

1.5 Solution Statement

An earlier competition from Jigsaw saw participants classify toxic/non-toxic comments, however, the models that were built back then fell short when it came to eliminate unintended bias, e.g. they gave a high likelihood of toxicity for comments containing certain identities (e.g. "gay"), even when those comments were not actually toxic (such as "I am a gay woman"). This happened because training data was pulled from available sources where certain identities are overwhelmingly referred to in offensive ways.[*cit.Kaggle*]

Key to this competition is to minimise this bias and evaluate a strategy able to classify with the same high level of accuracy samples belonging to different identities.

One solution could be the use of an **ensamble** of classifiers where each classifier is optimised on the classification of a particular identity. The likelihood of a comment being toxic would then be based on a weighted average of each single classifier where the weight given to each classifier is adjusted based on the identity of the message.

Not all data in the traning set will have been tagged with identities and therefore, in order to adjust the weight of the different classifier based on the comment's identity, there will be the need to predict the likely identity a of a comment.

This might require an initial classifier to first predict the possible identity of each comment, the prediction will be used as a feature for the ensamble of classifiers and used to calculate the weight for their score based on the predicted identity of the comment.

There will be different models suitable for classification and during the *Model Building* phase (discussed in Project Design section) I will have to evaluate the ML algorithm to use. Deep Neural Networks like CNN and LSTM are some that I'm considering to evaluate.

1.6 Benchmark Model

As any other Kaggle competition, submissions will be benchmarked against the test set held by Jigsaw producing a score as per the *Evaluation metrics* defined below.

Submissions to this competition must be made through Kernels, also the following conditions must be met:

- CPU Kernel <= 9 hours run-time
- GPU Kernel <= 2 hours run-time
- No internet access enabled
- External data, freely & publicly available, is allowed, including pre-trained models
- No custom packages enabled in kernels
- Submission file must be named "submission.csv"

1.7 Evaluation Metrics

This competition uses a newly developed metric that combines several submetrics to balance overall performance with various aspects of unintended bias.

Please refer to evaluation section of the competition and the provided benchmark kernel with code to calculate the competition evaluation metrics.

Here are defined the submetrics:

1.7.1 Overall AUC

This is the ROC-AUC for the full evaluation set.

1.7.2 Bias AUCs

To measure unintended bias, we again calculate the ROC-AUC, this time on three specific subsets of the test set for each identity, each capturing a different aspect of unintended bias. You can learn more about these metrics in Conversation AI's recent paper *Nuanced Metrics for Measuring Unintended Bias with Real Data in Text Classification*.

Subgroup AUC: Here, we restrict the data set to only the examples that mention the specific identity subgroup. *A low value in this metric means the model does a poor job of distinguishing between toxic and non-toxic comments that mention the identity*.

BPSN (Background Positive, Subgroup Negative) AUC: Here, we restrict the test set to the non-toxic examples that mention the identity and the toxic examples that do not. *A low value in this metric means that the model confuses non-toxic examples that mention the identity with toxic examples that do not*, likely meaning that the model predicts higher toxicity scores than it should for non-toxic examples mentioning the identity.

BNSP (Background Negative, Subgroup Positive) AUC: Here, we restrict the test set to the toxic examples that mention the identity and the non-toxic examples that do not. *A low value here means that the model confuses toxic examples that mention the identity with non-toxic examples that do not*, likely meaning that the model predicts lower toxicity scores than it should for toxic examples mentioning the identity.

1.7.3 Generalized Mean of Bias AUCs

To combine the per-identity Bias AUCs into one overall measure, we calculate their generalized mean as defined below:

$M_p(m_s) = (\frac{1}{N}\sum_{s=1}^{N} m_s^p)^{\frac{1}{p}}$

where:

MpMp = the ppth power-mean function
msms = the bias metric mm calulated for subgroup ss
NN = number of identity subgroups

For this competition, we use a pp value of -5 to encourage competitors to improve the model for the identity subgroups with the lowest model performance.

1.7.4 Final Metric

We combine the overall AUC with the generalized mean of the Bias AUCs to calculate the final model score:

$score = w_0 AUC_{overall} + \sum_{a=1}^{A} w_a M_p(m_{s,a})$

where:

- A = number of submetrics (3)
- $ms_{s,a}$ = bias metric for identity subgroup ss using submetric a w_a = a weighting for the relative importance of each submetric; all four w values set to 0.25

While the leaderboard will be determined by this single number, we highly recommend looking at the individual submetric results, as shown in this kernel, to guide you as you develop your models.

1.8 Project Design

Following are the high level steps I'm planning to take to build a toxic content classification algorithm:

1. Exploring data and collecting key metrics
2. Definition of evaluation metrics
3. Transformation
4. Model selection and building
5. Training, evaluation and fine tuning the model
6. Kaggle submission

1.8.1 1. Exploring data and collecting key metrics

First and foremost I will explore the dataset, during this phase I will gather various statistics like:

- Number of samples in the training set
- Distribution of toxic/non-toxic in the training set
- Distribution of toxic/non-toxic in the training set for each identity
- Number of samples with no identity set
- Distribution of toxic/non-toxic in the training set for each sub-group
- Number of words per sample
- Frequency distribution of words
- Distribution of sample length
- Most frequent words

1.8.2 2. Definition of evaluation metrics

It is important to have defined a method of evaluation from the very beginning and the Kaggle competition provides what are the evaluation metrics used.

However I will want to being able to calculate the score offline without the need to submit the code so I'll want to setup code that can calculate the evaluation metrics offline.

1.8.3 3. Transformation

The dataset will need to be transformed before it can be submitted to a model, this requires the removal of stopwords, identification of punctuation (with consideration of punctuation when used to obfuscate offensive words), lemmitisation and tokenisation. Lastly I will transform words into embeddings evaluating libraries like **Word2Vec**, **GloVe** and **FastText**.

One thing to note is that Word2Vec and GloVe only learns vectors for completed words found in the training corpus which means that words that are misspelled or that are camuflaged using special characters like $#!* won't be assigned a vector.

On the other hand FastText breaks words into several n-grams (sub-words). For instance, the tri-grams for the word apple is app, ppl, and ple. Rare words can now be properly represented since it is highly likely that some of their n-grams also appears in other words. [*cit.Medium*]

1.8.4 4. Model selection and building

An earlier competition from Jigsaw saw participants classify toxic/non-toxic comments, however, the models that were built back then fell short when it came to eliminate unintended bias, e.g. they

gave a high likelihood of toxicity for comments containing certain identities (e.g. "gay"), even when those comments were not actually toxic (such as "I am a gay woman"). This happened because training data was pulled from available sources where certain identities are overwhelmingly referred to in offensive ways.[*cit.Kaggle*]

Key to this competition is to minimise this bias and evaluate a strategy able to classify with the same high level of accuracy samples belonging to different identities.

One solution could be the use of an **ensamble** of classifiers where each classifier is optimised on the classification of a particular identity. The likelihood of a comment being toxic would then be based on a weighted average of each single classifier where the weight given to each classifier is adjusted based on the identity of the message.

Not all data in the traning set will have been tagged with identities and therefore, in order to adjust the weight of the different classifier based on the comment's identity, there will be the need to predict the likely identity a of a comment.

This might require an initial classifier to first predict the possible identity of each comment, the prediction will be used as a feature for the ensamble of classifiers and used to calculate the weight for their score based on the predicted identity of the comment.

There will be different models suitable for classification and during this phase I will have to evaluate the ML algorithm to use. Deep Neural Networks like CNN and LSTM are some that I'm considering to evaluate.

Convolutional neural network are suitable when data has positional information.

Long Short Term Memory neural network, like RNNs are designed to use sequential data, when the current step has some kind of relation with the previous steps and they are designed to remember things in the long term. [*cit.Quora*]

While sentiment analysis might not need information of position of words, I believe considering the position of words will improve the accuracy across all the different identities and will help with unintended bias (e.g. "is an offensive word being use to quote someone else's comment" or "is the tone demeaning without the use of any particular offensive word" etc.)

A further consideration is the Kaggle kernel limits, it imposes a CPU Kernel <= 9 hours run-time and a GPU Kernel <= 2 hours run-time. This limitation will be a guiding factor when deciding what algorithm to choose.

1.8.5 5. Training, evaluation and fine tuning the model

Having defined a suitable model, probably the hardest part will be evaluation and fine tuning of the model. Techniques like Grid-search can help choose more suitable hyperparameters however this process is painstakingly long.

1.8.6 6. Kaggle submission

It might become tempting to submit to Kaggle frequently to evaluate the performance against Jigsaw's scoring and optimise the model against this score.

There is various evidence that this is not a winning strategy, in fact there is a chance of slowly overfitting the model on this test set, it might become a good predictor of the test set but it will not be able to generalise. [*ref.Kaggle*]

As such, I shall refrain from making too frequent submission or becoming fixated on the Kaggle leaderboard score.

1.9 Other Resources

- BERT: Pre-training of Deep Bidirectional Transformers for Language Understanding
- Sequence classification with human attention
- Language Learning with BERT - TensorFlow and Deep Learning Singapore
- Abusive Language Detection with Graph Convolutional Networks
- Bag of Tricks for Efficient Text Classification
- The Dangers of Overfitting or How to Drop 50 spots in 1 minute
- Google SentencePiece
- The Illustrated BERT, ELMo, and co. (How NLP Cracked Transfer Learning)