Recurrent Neural Networks and Artificial Neural Networks for Text Classification

With the recent exponential growth and usage of Natural Language AI, distinguishing between human and machine generated text has become increasingly harder. Therefore, it has become relevant and important to utilise machine learning strategies to address this issue. In this project, we have decided to investigate the proficiency of Neural Networks in classifying text as human or machine generated text. We will explore Recurrent Neural Networks (RNN) and Artificial Neural Networks (ANN) in the following analysis.

Datasets

For this project, we were given two sets of training datasets. The first dataset contained 125k instances of tokenised text labelled either as machine or human generated, attached with additional information on machine_id. The second dataset contained 500 similarly structured instances. After examining both datasets, it was clear that the main challenge of this project was working with imbalanced datasets. Dataset 1 contained significantly more human generated instances, whereas dataset 2 contained more machine generated instances. An additional challenge of training machine learning algorithms using dataset 2 was that it contained significantly less instances.

Pre-processing

Feature Engineering

In order to add less weighting on less meaningful words and potentially gaining some insight into what specific words are more commonly used by machines in comparison to humans, we chose to use a word vectorisation method called Term Frequency - Inverse Document Frequency (TF-IDF). Term frequency calculates the proportional frequency in which a word appears in a text instance, while inverse document frequency calculates the inverse of the proportion of text instances that contain that word. While term frequency captures the importance of a specific word in a single instance, the inverse document frequency penalises words that appear in too many instances as they are less likely to be meaningful. An important consideration when using this vectorisation method is that this significantly increases our feature space, as an additional feature was created for each unique word token in each dataset.

Balancing Datasets

In order to improve the balance of dataset labels in both domains, we have used both oversampling and under sampling methods. As the detection of machine generated text was our main goal, we decided not to alter the number machine samples to avoid the loss or tampering of machine information. Instead, we balanced the datasets by altering the number of human samples. Though we were not able to achieve a 1:1 label ratio due to the severity of skewness in the original data, the ratio was significantly improved. The results are as follows:

Domain	Balancing method	Original label count	Revised label count
1	Under sampling	3500 machine labels 122584 human labels	3500 machine labels 99999 human labels
2	Oversampling	400 machine labels 100 human labels	400 machine labels 200 human labels

Methodology

Our Overall Approach

Due to the differences in the data in domain 1 and 2, we decided to adopt two different approaches in training our models which was later used in prediction of our final test set. Our approaches are summarised as follows:

Domains	Model used	Ingestion data type	Information learnt
1	RNN	Word sequences	Types of sentence structures used more frequently in machine generated text
2	ANN	TF-IDF vectorized words	Meaningful words used more frequently in machine generated text

We decided that domain 1 data was appropriate for approach one, as domain 1 contained more data. This means that it was more likely that complex patterns such as text sequences could be learnt. Due to domain 1 being more heavily imbalanced, it was also necessary to use a more complex model to compensate for this. Domain 2 data was used for approach 2, as learning from word frequency was achievable with less data.

Model Building for Domain 1 - RNN

Our aim for the first model was to build a model capable of learning text patterns as sequential data in order to predict whether text was machine generated based on sentence structure. To do this, we used RNN as this was a model capable of learning complex patters from sequential data.

The first layer of our RNN model consisted of an embedding layer which takes high dimensionality input and maps it to a lower dimensional space for fast processing. This layer is also capable of learning relationships between inputs. A challenge that we faced with this layer was the dynamic input of our data. Due to the varying lengths of the text in different instances, we were not able to provide the embedding layer with a fixed number of nodes. In order to solve this issue, we decided to pad shorter texts using 0 values, and create an upper bound on the number of words within each instance. The next layer of our RNN model was a simple RNN layer. The purpose of this layer was to create a bidirectional flow of data between the input and output wherein the output values were able to be fed back into the input for the model to learn patterns present in the sequences of words. The final layer of our model was a dense output layer with the activation function as the sigmoid function. This activation function allowed us to assign an instance a value between 0 and 1, representing the probability that it was machine generated. From here, we were able to set a probability threshold to classify the labels.

Parameter tuning

The parameters to tune in this model included the upper word bound (input size for first layer), output size for first layer and probability threshold for classification in the final layer. As our goal was to identify machine generated text, the metric we used in tuning our parameters dataset was the precision of machine labels, as it favoured distinguishing machine labels over human labels. This was especially important as our data was heavily imbalanced and skewed towards human labels. After tuning these parameters, we obtained these evaluation metrics:

Accuracy	Precision	Recall
97%	70%	44%

Other Approaches

Another approach we explored was fitting domain 1 data to a random forest model. However, this model was computationally very expensive and performed poorly. This is likely due to the high imbalance of data that random forest models are ill-equipped in dealing with. Due to the text data being in tokenised form, we were also unable to properly pre-process the text into features suitable for a random forest to run with our limited computational power. We decided not to utilise machine_ids from the domain 1 dataset as it was already heavily imbalanced against machine labels.

Predicting machine_ids would have caused a heavier imbalance in data and would likely have performed poorly.

Model Building for Domain 2 - ANN

Our aim for the second model was to build a model capable of learning what meaningful words are used more frequently in machine generated text than in human generated text. To do this, we used an Artificial Neural Network as this was a model capable of taking high dimensional vectorized data.

We used two dense layers to train our model and then a final output layer to predict the validation as well as test instances. Both the dense layers were built by setting them up to ingest the TF-IDF data and produce the required output. The ANN model being a feed-forward model, did not include any back propagation to learn from the previous sequences of words. The model was straightforward without any major obstacles due to the sample size of domain 2 being considerably smaller compared to domain 1.

Parameter tuning

The parameters to tune in this model included the batch-size and the number of units for each of the two hidden layers. As our goal was to identify machine generated text, the metric we used in tuning our parameters for this dataset was accuracy of machine labels, as it reflected how well our model performed in classifying both the instances correctly. After tuning these parameters, we obtained these evaluation metrics:

Accuracy	Precision	Recall
93%	90%	100%

Other Approaches

Simpler models such as a Support Vector Machine was considered to perform the same task. However, this did not perform well in classifying with an imbalanced dataset, as it was heavily biased towards the majority class. Additionally, due to the similarity in the machine and human generated text, it is possible that the two classes were not easily separable, even with high feature dimensionality. In order to understand whether our model can have a higher learning rate by separating our training data by machine_ids, which we would then map into machine and human labels, we fit a naïve bayes classifier for the same. However, this model performed poorly during evaluation. This is likely because we took a naïve approach in assuming the distribution of the machine_ids.

Final Prediction

Our final testing data consisted of 1000 instances of text that we then classified as either human or machine generated. We predicted the first 600 instances using the RNN model trained on dataset 1, and the last 400 instances based on the ANN model trained on dataset 2.

Further Investigation

In the future, we would recommend further investigating the distribution of the machine IDs, as it would likely lead to a higher learning rate by classification models and would allow the training data to be fitted more accurately. We would also recommend using more complex techniques in balancing the data, as it would lead to less skewed models. Given one of our limitations being not able to perform thorough hyperparameter tuning on our neural networks due to our limited computational power, it would also be worth exploring this further.

https://www.analyticsvidhya.com/blog/2020/02/cnn-vs-rnn-vs-mlp-analyzing-3-types-of-neural-networks-in-deep-learning/
https://monkeylearn.com/blog/what-is-tf-idf/