Training with Mislabeled Data and Improvements to Convolutional Text Classification

Darrell Aucoin (1003562316)

August 10, 2018

1 Company: Datalogue

Datalogue is a data preparation company streamlining and automating the ETL (Extract Transform Load) pipeline. This an essential task in wrangling data from multiple sources for analysis and insights. Currently, Datalogue is concentrating on ontological mapping, segmentation analysis, and field standardization and recognition.

2 Supervisors

Name	Role	Email
David Duvenaud	Academic	duvenaud@cs.toronto.edu
Radu Craiu	Academic	craiu@utstat.toronto.edu
Nicolas Joseph	Industry	nicolas@datalogue.io
Bryan Russett	Industry	bryan@datalogue.io

3 Role in Company

My title in Datalogue over the internship was Machine Learning Research Associate. The basic role was to improve and/or create new machine learning models to be used as the engine for various tools that Datalogue either currently provides or is looking to provide.

Ownership of ideas and concepts is big in Datalogue. There were many times I suggested changes we could make to the existing machine learning models or new models and I was to implement these changes, experiment, and talk about the results. When there were no pressing issues, my day to day tasks could be very open, effectively looking for ways to improve existing machine learning models or create new models that would be of use to the company. In the context of the internship, I was applying my research topic of improving machine learning when dealing with mislabeled data, using the data used for training at Datalogue. The second part of the research was based on various ideas and experiments that I thought up and conducted when learning the existing machine learning models Datalogue had in place.

Outside of this primary task I also collected more data for training Datalogue's machine learning models, trained models for clients, and cleaned data. Cleaning data was also done in relation to working on mislabeled data.

4 Role and Expectations

The original description of the job was to conduct research on ways of automating data preparation. This includes enhancing existing models, applying research to the work at Datalogue, and prototyping models. Since this was a startup, a strict description of the job was not viable as a more exact job description would need to be changed as Datalogue's direction changes. I expected that the job would also include some data preparation, including data cleaning as it is a fundamental aspect of most Data Science jobs. The amount of time I did on data cleaning was more than I initially anticipated, but that was partially in relation to my research project on mislabeled data. Although this data cleaning was the most tiring aspect of the job, it also had the most impact on the performance of the models.

In the job description they specified that the research project will be using ConvNets, Long Short Term Memory networks (LSTMs), and Conditional Random Fields (CRFs) for field recognition and ontological mapping. For programming, the job description mentioned that the programming languages python, scala, and rust along with the packages Tensorflow and MXnet would be used. This matched fairly well with what I experienced, except that scala was only needed sparingly and rust wasn't used at all for my work.

In the way of support, my initial expectations in the beginning were that the Montreal office would have another Machine Learning Researcher who I could learn from and that my industry supervisor(s) would be in Montreal with me. When I first started there was another Machine Learning Intern in the Montreal office, with the rest in New York City. My industry supervisor(s) were primarily located in New York City as well, with the primary means of communication being video chat and them coming into Montreal once every months or so. My fellow Machine Learning Intern was essential for adjusting to work at Datalogue, allowing me someone to ask questions about the existing models and code, as well as bounce ideas off of.

The work environment is much more relaxed than I initially expected, only pushing for more work hours when a due date for a large deliverable was on the horizon. There was also the freedom to explore new models and try new things, usually with only the need of a discussion with the rest of the Machine Learning team. Everyone was always approachable, and ready to answer questions given they had the time to do so.

5 Outcomes Achieved

Since the start of my internship, the text classifier at Datalogue has gone from 94% validation categorical accuracy to 98.7%. This is a substantial improvement, especially since I also broke some of the original classes into more refined classes. For instance, originally first names, last names, and full names were labeled under one class called personal name and I broke them down into first names, last names, and full names with only a slight impediment to their class precision and recall scores. This overall performance increase was in response to data cleaning, data gathering, and the improvements to the text classifier that I implemented. Although, the majority of this increased performance is due to the data cleaning and gathering I performed, these improvements also contributed to this performance increase and is currently integrated into Datalogue production. My work on mislabeled data is currently inconclusive, the model worked well with toy datasets like MNIST, however with Datalogue data's the model performance didn't exceed the performance when just training on the mislabeled data. I also started work on a new classifier for columns in a table. Only preliminary work has been done so far, but initial tests showed a validation accuracy of 99.9%.

The text classifier works by mapping characters in the input text to trainable tokens in the neural network, my improvements relate to changing these mappings slightly. Originally, characters that were not used very often were mapped to the unknown token, instead I changed the mappings so they were mapped to their closest equivalent that we did have a token for. For instance, 'é' might now be mapped to the 'e' token instead of the unknown token. Another improvement was adding a 'beginning of term token to the mapping (a 'end of term' token was already used). The text ontology classifier works by identifying patterns in its input tokens. With 'beginning of text' and 'end of text' tokens, it can associate patterns with where they are located in the text and thus increasing classifier performance. All of these improvements have been integrated into Datalogue's product.

For the work on mislabeled data, the objective was two fold: increase the performance of a classifier when some of the data is mislabeled, and help identify mislabeled data for cleaning. The system proposed works by the neural networks having two components: a classifier, and a model of the noise (mislabeled data). After training, this model of the noise could theoretically help find the true label again, called the denoise (or denoising) model. On some publicly available datasets like MNIST (after corrupting the labels randomly), there are some definite results improving the classifier performance compared to the classifier just trained on the corrupted labels as well as identify either mislabeled items (table 1). However, for the data at Datalogue the results were less conclusive. One of the problems of measuring the change in performance with Datalogue's text data is that Datalogue's text data is not fully cleaned, which means that measuring the performance on the true labels requires extensive data cleaning. However, even after some data cleaning, the performance of the classifier after training accounting for the noise (mislabeled data) did not out perform the classifier just trained on the mislabeled data.

Corruption	Classifier	Noise Matrix	Logistic Regression Into Noise Matrix	Last Hidden Layer Into Noise Matrix
0%	98.99%	98.99%	98.93%	99.02%
(denoise)	(N/A)	(99.95%)	(99.94%)	(99.96%)
12%	98.70%	98.98%	98.89%	98.77%
(denoise)	(N/A)	(99.73%)	(99.71%)	(99.72%)
33%	98.48%	98.78%	98.94%	98.89%
(denoise)	(N/A)	(98.90%)	(99.49%)	(99.50%)

Table 1: Noisy layer experimental results on MNIST dataset. Corruption of labels are done randomly.

For identifying mislabeled data, I proposed finding the prediction value of the label for the data point over all data points in the dataset. After sorting these data points by this prediction value in ascending order, we now have a list of data points where the data points on top of the list have a low probability of being the class its labeled as. In this area, the work has produced results. This technique of sorting on prediction value definitely seems to help find mislabeled items as can be seen in table 2, and this works even better when we model the noise (mislabeled data). For the top 1000 entries, for all models considered this technique brought the mislabeled data points more to the surface with the best model being the classifier after modeling the noise. In those mislabeled items, the majority of the predictions from the model were correct.

Model	Accuracy on True Class	Accuracy on Label	Top 1000 Corrupt Entries	 Correct Predictions
Classifier trained on mislabeled data	98.89%	98.91%	316	276
Denoising Model	99.89%	99.93%	521	488
Classifier after modeling noise	97.61%	97.62%	629	519

Table 2: Accuracy and count of mislabeled data of top 1000 data points.

6 Research Incorporated

Datalogue's text classifier is based on the paper "Very Deep Convolutional Networks for Text Classification"[1], which describes a text classifier using convolutional neural networks. This is the main classifier model that I worked with over the internship. This model worked by mapping characters in a string to a set of tokens that are trained on the neural network. As neural networks require a fixed input size and shape, when the length of the string was less than some fixed length, padding character tokens were added to fill up the space. It was also standard convention to add a end of term token after the end of the input string to delimit the end of the string.

A convolutional layer runs over these tokens, identifying patterns of characters that are 3 tokens long. This then goes through another convolutional layer, essentially finding patterns of patterns and so forth. My improvements that I purposed related to how the characters in the input strings mapped to the tokens used.

My work on improvements on convolutional neural networks for text classification is based around this paper and the questions I had on why they decided on making the decisions that they did in order to understand the model. For instance, in the paper they described the set of characters they mapped tokens to, including padding, space, and unknown tokens. Every character that is not in this set of characters is mapped to the unknown token. This seemed like a waste to me as many of these characters that would be mapped to the unknown token are very similar to one of the characters inside this set of mapped character tokens. I then modified the mappings so that characters outside the character token set were mapped to their nearest character equivalent if possible. This helped improve the performance of the classifier by mapping less characters to the unknown token.

The inclusion of the beginning of term token came about by realizing that the patterns that convolutional neural networks find are by the nature of convolutional networks irrespective of their locations in their input. However, for the text Datalogue is trying to classify the patterns usually just occur in one location in the data. By adding the beginning of term token, the neural network can include this token in its pattern recognition and thus can place the pattern identification at the beginning of the text if it's helpful in classification.

The work on mislabeled data model is based around the work by S. Sukhbaatar[2] and N. Natarajan[3] as well as a course project I did in Winter 2017. The model has 2 parts: a classifier ($p\left(t \mid x\right)$) and a model of the mislabeled data ($p\left(c \mid t\right)$) where x represents the input, t the latent true label that is not seen, and c as the corrupted label in the dataset. From this we can create a model to train on the true label t without observing it:

$$p\left(c \mid x\right) = \sum_{t} p\left(c \mid t\right) \times p\left(t \mid x\right)$$

My contribution is that by rearranging the equation we can get $p\left(t \mid c, x\right)$ which should give a higher predictive accuracy of the true labels, t. This can then be used to help identify entries that have been mislabeled (or are outliers for the model).

$$p\left(t \mid c, x\right) = \frac{p\left(c \mid t\right) \times p\left(t \mid x\right)}{\sum_t p\left(c \mid t\right) \times p\left(t \mid x\right)}$$

For each data point x and corrupt label c, we can get the probability of t for each data point entry. After sorting (in ascending order) the dataset according to $p\left(t = c \mid c, x\right)$, we now have a list of data entries in which the first entries are more likely to be either mislabeled or outliers for the model.

The noise model, $p\left(c \mid t\right)$, above assumes that corruption is only dependent on the true label and independent of the input x, we can remove this assumption give a mislabeling model of $p\left(c \mid t, x\right)$. That is the probability of misclassification given the true label and it's associated input x. For modeling $p\left(c \mid t, x\right)$ two main models were tested, one using the multiclass logistic regression of x for every true value t, and another that takes the output of the last hidden layer of the classifier rescaled and feeds that into the model of the mislabeled data.

[1] Conneau et al., "Very Deep Convolutional Networks for Text Classification." 2016.

[2] S. Sukhbaatar et al., "Training convolutional networks with noisy labels," 2014.

[3] N. Natarajan et al., "Learning with noisy labels," in Advances in Neural Information Processing Systems 26, 2013, pp. 1196–1204.

7 Impact on Company

I had a good impact on Datalogue over the last 8 months. Several of my ideas for improving Datalogue's text classifier have been integrated into production. This along with data cleaning has increased text classification accuracy from 94% to 98.7%, which is a substantial improvement. Currently, the work with mislabeled data is not ready for production, other than possibly to help find mislabeled data points. However, if this is refined more, it could dramatically help improve not only internal Datalogue modeling, but client-facing modeling as well.

There are also model ideas that I came up with near the end of my internship, that while not completely fleshed out could become a part of the next set of products for clients. For instance, I also created a new model for column classification, which takes in many data points from the same source (i.e. a column from a table) which ideally should have the same class and classifies the group of data points. Only preliminary work has been done so far and trained only once with Datalogue's data, however this gave a 99.89% validation accuracy. The current plan for this is to combine this model with another model that takes in information from the header to give a classification for the column.

8 Lessons Learned

I learned several lessons over my internship. I gained more insight into how convolutional neural networks work with text: how it finds patterns and then patterns of patterns. I learned more on good formation of code and its importance in a team environment, although I still have much more to learn. I learned that I should document more of my actions and experiments I run, and I will aim to do so in the future.

In the area of mislabeled data, I learned the importance of a clean dataset on the performance of classifiers. The amount of mislabeled data points had a much stronger performance degradation impact on the Datalogue's text classifier over the MNIST data and classifier. Currently, I believe this is due to how similar the data points for different class are to each other with each classifier, as well as the dimensionality of the input data (text is 1 dimensional while images are 2 dimensional). When I first applied my model for dealing with mislabeled data to Datalogue's text data, the validation accuracy was close to 50%. After cleaning the dataset a bit (specifically looking at data points that have multiple labels and removing erroneous labels) and applying the model again the validation accuracy jumped up to over 90%. I believe these two issues made the model for dealing with mislabeled data difficult to work with and allowed the text classifier to learn the wrong lessons for classification.

An important factor for robustness against mislabelling is how different the classes are to each other. To illustrate this, consider the data points as points in some space where the space is defined as points close to each other have similar structure to each other. As the classes become more linearly separable (the different classes have very different patterns from each other) then the division between the classes become more easily defined. When mislabeled data points are introduced, it will try to incorporate these data points in the classification division. If the classes are highly linearly separable then there are a lot of classification divisions that would still work fairly well, however when the classes are not very linearly separable then finding a good classification division becomes much more difficult. Datalogue's text data is not very linearly separable for several of it's classes, for instance there are many data points which can have multiple labels, such as the string 'David' could be a first name or a last name. Before I started data cleaning, there were many data points that multiple erroneous labels and after a round of data cleaning removing these erroneous labels, the mislabeled data model had the significant jump in validation accuracy, even though the number of data point labels changed was less than 5% of the dataset.

It is also likely that dimensionality of the data affects how robust the performance of a classifier is against mislabeled data. Text is 1 dimensional (it's a sequence of characters) while images are a 2 dimensional series of values. Higher dimensional data allow for a greater variety of patterns for the neural network to be to identified and associated with the different classes.

9 Next Steps

I have been offered and accepted a full-time position at Datalogue, as such the next steps in my career will be with Datalogue. Within Datalogue, I will be working on column classifier and other machine learning models that will hopefully be of use to Datalogue. I will also try to see if further progress can be achieved with working with mislabeled data. Although the results aren't satisfactory yet, there are a few more ideas I would like to try out. For instance replacing the soft-max layer of the neural network of the text classifier with a SVM (support vector machine) layer, this might help with mislabeled data as SVMs are more robust against outliers (the classifier would see mislabeled data points as outliers).

I will also try to keep up with research and see how they can be applied to the needs of Datalogue and it's clients. For instance, I'm interested in the new idea from Geoffrey Hinton about Capsule Networks and when I can see how they can be applied to text classification. Datalogue encourages going to conferences, so I will be going to them when I can.