**Prediction Churn Reduction**

Bhantana H., Bizhanov D., Harutyunyan T., Kishore S.

**Motivation**

The role of machine learning has been drastically increasing in the last decade. It is hard to imagine the modern world without this technology anymore. Machine learning algorithms are used ubiquitously in many areas. For instance, banks use predictive algorithms to score their clients and assess risks. IT giants such as Google and Amazon use complex Deep Neural Networks as essential components of their services - Google Translate, Alexa, Google Photos, self-driving cars, and others. As the digital component of our lives is becoming dominant, the importance of cybersecurity becomes more prominent. Being an extremely flexible tool, machine learning is used in many different contexts starting from simple spam filters and ending with sophisticated methods to detect suspicious activity which helps to spot attacks and block any malicious intent. The reasons why machine learning and deep learning are successful in such a dynamic field are manyfold: automated algorithms are better adapted for rapid changes in the virtual environment than heuristic rules based on limited experts’ experience, expert-based systems are not scalable, it is easier and faster to tune a machine learning algorithm than to develop a brand new set of rules.

Despite the obvious usefulness of machine learning algorithms still, many challenges need to be overcome. One such challenge is prediction churn. Prediction churn inevitably occurs during the iterative lifecycle of any predictive algorithm. Intuitively it is a discrepancy between the predictions made by the same model trained on different data samples and can be defined as a proportion of mismatched data points to the number of overall predictions. This mismatch may create many problems for the end users and potentially can be detrimental for businesses that use such algorithms. As an illustration, many people are familiar with the auto-classification of their photos. After uploading one image, a model predicts that it is a tree. However, after retraining this model on an extended data set, prediction churn may lead to a different classification, let's say a car. Even though the above example is not dangerous, it led to a scandal in one of Google's services. In a different context, this undesired behavior could lead to serious negative consequences.

Prediction churn can be further split into two categories. The first category is desirable or good churn. It happens when an updated version of the model makes correct predictions in cases where the old model makes mistakes. The second category is undesirable or bad churn. This churn is the most problematic since an updated model makes mistakes in cases where the old model’s predictions are correct. In an ideal setting, our goal is to reduce undesirable churn and at the same time increase, accuracy which means that increasing the former churn category is not only not a problem but a beneficial side-effect. It is worth mentioning that there is an alternative definition of prediction churn that is based on the notion of divergence. We find that the latter definition is not as intuitive as a proportion-based one. In addition, several variants of derived metrics exist in the literature. For instance, churn ratio, and win-loss ratio. We will further elaborate on these metrics in our methodology section.

Our client for this capstone project is Proofpoint. Proofpoint is an American enterprise company that is focused on cybersecurity. They cover a full range of protection including cloud accounts, emails, web security, and private data safety. Proofpoint has a massive set of machine learning models that work in the production environment. Given the delicate area in which Proofpoint operates and the large scale of production models, our client faces many challenges including prediction churn. Because of data sensitivity and other privacy restrictions we do not have access to their models or their data. Therefore, in order to achieve our goal we will use well-known benchmark data sets that many researchers have been using in the machine learning field.

Despite its significant importance, prediction churn is a highly overlooked research area. There are several known approaches for churn reduction. The first approach is based on changing training methodology and is highly focused on training stability since one of the sources of prediction churn is random initialization, different training runs, choice of hyperparameters, and random order of GPU computation. Thus, when such randomness is reduced, prediction stability increases and prediction churn reduces as well. However, this path is not feasible for us since our initial conditions include black-box model treatment. The black-box model treatment assumes that we know nothing about the underlying prediction model. This property is one of the requirements made by our client. The second approach frames the problem from a label modification perspective. Instead of training on the true labels, the target model trains on a convex combination between hard one-hot encoded labels and soft labels produced by the auxiliary model. During our literature review phase, we selected three papers to focus on: Launch and Iterate: Reducing Prediction Churn, Locally Adaptive Label Smoothing for Predictive Churn, and Churn Reduction via Distillation. The importance of these papers lies in the experimental evidence that the methods described can reduce churn and the methodologies satisfy our constraints and follow the black-box models paradigm.

Knowledge distillation has been around since 2015 and has been used in many surprising contexts. Originally the main goal of the method was to decrease the complexity of deep learning models by introducing a teacher model. Having trained a teacher model, a student model which is usually much simpler learns from the first model by parroting predictions. In our context, a student model learns more information from soft labels than from hard ones. Introducing soft labels creates meaningful geometry in the label space and gives richer vector representation and useful attributes such as similarity, distance, and angles. The authors of Churn Reduction via Distillation prove that knowledge distillation is equivalent under mild assumptions to constraint churn optimization. Since the constraint optimization approach is more involved, churn reduction using distillation is the first candidate for implementation. The main limitation of this method is that it can not be used with the classical machine learning models and it is bound to neural networks.

The authors of Launch and Iterate use the anchor method. Chronologically this paper was written first and from a technical perspective, it is more involved. Since this paper was one of the first on the subject, the authors rigorously defined churn for the first time. In addition, as the anchor method employs Markov Chain Monte Carlo (MCMC) method and two stabilizing operators this method can be used with a wider range of machine learning models.

The third and the last method that we have selected is based on locally adaptive smoothing paper. The authors use the K-Nearest Neighbors algorithm instead of a teacher model for finding similar data points using the logit layer of a neural network that is used as a helper model. Further, they combine local and global smoothing to produce modified labels. The result is similar to the first two papers and we decided to implement this method after the first two.

It is worth mentioning that the three papers use different data sets and metrics. Therefore, it is hard to make conclusions about the methods’ superiority. After we implement three methods, we will compare them on the same sets of data using the same metrics using a standardized experimentation protocol. Thus, experimentation is a crucial part of our project since it gives us a deeper understanding of the current methods and provides insights. The knowledge acquired by conducting experiments will guide us in our next phase - package creation. Proofpoint is interested in a model-agnostic framework that they can use on a daily basis. This requirement adds an additional layer of complexity as we need not only a working package for one popular machine learning library but an easily extensible package with a plugin-based API. Preliminary analysis shows that the most popular deep learning packages of today are PyTorch and TensorFlow. In addition, there are dozens of different machine learning packages such as Scikit-learn, Xgboost, FastGBM, and others. Our client may use any of these and potentially packages that do not exist at the moment. Therefore, we need to develop an interface that will allow adapting all possible models with the least effort.

Our final and most challenging goal is to develop a novel method for churn reduction. At this point it is hard to foresee the outcome, however, working on the current methods and developing the package should give us a strong foundation for further advancement.

**Project milestones**

The capstone project consists of three milestones. First, we need to create a unified approach to conduct our experiments, implement three methods that we covered in the motivation section, and make comparisons and conclusions based on the results. Second, we will create a python package and make the methods from the first milestone generic and independent of underlying models. That will ensure that our clients will be able to use the package in their environment. Finally, we will work on a novel churn reduction method. We will incorporate knowledge gained from the previous two steps.

**Experiments**

To conduct a thorough experiment, we will implement knowledge distillation, anchor, and label smoothing methods on different classification models using various data types. Namely, ResNet for image classification, Bidirectional LSTM for sentiment analysis (text classification), and TabNet for tabular data classification. The data sets we have selected are

CIFAR 10 (Canadian Institute for Advanced Research), Online News Popularity, and IMDB reviews. CIFAR 10 is a collection of images 32 by 32 pixels. This data set contains 60000 samples and 10 classes. Online news popularity is a tabular data set with 40000 samples, 60 features and 2 classes. Imdb online review is textual data with 2 prediction classes and it contains 60000 reviews. We have chosen these particular data sets for the following reasons. First, we do not have access to our client’s data as we mentioned earlier. Second, these data sets are widely used in the machine learning community and in research. Finally, the data volume is large enough to train neural networks and satisfies our computation capacity.

The models we will employ have sensible default hyperparameters in place. Having consulted with our client we have come to a decision not to conduct a hyperparameter tuning at least during the initial stage of our experiments. Later on we might optimize hyperparameters of student models since our client has pointed out that the main focus should be on these models. If we use hyperparameters tuning we will employ the Ray framework.

In order to compare churn reduction methods we have selected the following four metrics: churn, churn ratio, win-loss ratio, good and bad churn. *Churn* is the expected amount of disagreements between two models, i.e it is the average number of the labels misclassified between student and teacher. *Churn Ratio*C(f0,f1) / C(f0,f2) where the old model is f0, the new model trained with a methodology is f1, and the new model trained without any methodology is f2. This is a measure of model improvement, where a lower ratio indicates that the model methodology reduces churn. *Win loss Ratio (WLR)* where the WIN is when the new model is able to correctly classify data points that the old model misclassified. LOSS is when the new model misclassified data points that were correctly classified by the old model. *Good Churn and Bad Churn***,** this metric divide churn into two categories: good churn—when the teacher's model predicts wrong, but the student's model predicts the correct classes; bad churn - when the predictions of the teacher's model are correct, but the student's model predicts the wrong classes.

The experiment will be conducted in the Colab, using GPUs. In order to ensure reproducibility of the experiment, we will log the details of the experiments with MLFlow.This tool allows logging all metrics with parameters as well as artifacts such as models and data used for training.

**Package development**

The main goal for this milestone is to create a python package. As mentioned before, the package has to be general and model agnostic, meaning that the teacher and student models can be any python model. The process of creating the package will require a few steps: software design, developing core methods compatible with PyTorch framework, extending the package to TensorFlow, and provide interface for further extensibility.

In the Software design stage, we will deliver architectural structure. We will use UML diagrams for documenting interclass connections. During this phase, we will restrict ourselves to the PyTorch library and make sure that the package works as expected. In order to make that happen we will develop a set of unit and integration tests using the Pytest framework along with the PyTorch internal testing functionality.

We are planning to extend the package to be compatible with TensorFlow in order to ensure that the solution covers more use cases. During this phase, we need to find differences and similarities between the two frameworks. It may require some refactoring in the initial version of the package.

Finally, to make our framework truly general we need to develop an extension-based API and provide a code interface. This step will guarantee that new frameworks will be compatible with our code base.

**Work on a novel method**

The final milestone is the development of a novel prediction churn reduction method. After we have worked on the existing methods and package creation we will have a strong foundation to start working on a new churn reduction approach. This milestone will be divided into sprints at the beginning of the second semester since we do not have enough information to develop a plan of actions at this stage.