CS 726

Project Report

Training a Generative model for Weak Supervision

Team Members

Roll Number	Name
190100064	Kaustubh Shripadraj Ponkshe
19D070052	Sheel Shah
19D070048	Puranjay Datta
19D180009	Deshpande Hrushikesh Shriniwas

Instructor : Prof. Sunita Sarawagi

Contents

1 Task Description

Deep learning is a rapidly evolving field that is driving advances in a range of applications such as image and speech recognition, natural language processing, and robotics. In many deep learning applications, having large amounts of labelled data is crucial to achieving high performance. However, obtaining and labelling data can be a tedious, time-consuming, and expensive process. However, data programming, a technique introduced by Snorkel, can help programmatically label data in large amounts. This is done by constructing labelling functions for weak supervision which can be achieved by training a generative model to learn the true labels as latent variables and the weak labels as their noisy samples. These labelling functions can be constructed using a range of techniques, including expert rules, distant supervision, and heuristics. By training such a generative model, we can create a discriminative model (on top) that generalizes the training data. Additionally, we extend and generalize the data programming paradigm to learn a model for a regression type or vector output labelling task.

For instance, consider the task of sentiment analysis on reviews. 'Weak' labels for whether a review is positive/negative can be obtained via human heuristics and/or NLP models. Consequently, the true labels can be learnt as a distribution, given the weak labels. We further demonstarte how these probablistic labels can help generalize a deep learning based discrimantor beyond just labelling function heursitics.

2 Challenges addressed

2.1 Motivation

Machine learning has become an essential tool in a variety of applications, from computer vision to natural language processing. One of the key challenges in deploying machine learning systems is the need for large amounts of labelled data, which is often a costly and time-consuming process. In many cases, the availability of labelled data can be a bottleneck, limiting the effectiveness and scalability of machine learning systems.

To address this issue, researchers have developed a novel system called Snorkel, which allows users to train machine learning models without the need for hand-labelled training data. Instead, Snorkel enables users to write labelling functions that express arbitrary heuristics, which may have unknown accuracies and correlations. These labelling functions can then be used to programmatically label large amounts of data, effectively bypassing the need for hand-labelling.

Snorkel incorporates an end-to-end implementation of data programming, which enables it to denoise the output of these labelling functions without relying on ground truth. This approach is particularly useful in situations where the availability of labelled data is limited or where the cost of labelling data is prohibitively high.

In the case of sentiment analysis, one has access to the various textual data available, however manually labelling such data is a humongous task.
Ideally, the labels from many weak supervision sources can be combined to increase the accuracy and coverage of the training set. However, the challenges in doing so effectively are:

- There is likely to be overlap and conflict among sources of data, and it is necessary to estimate their accuracies and correlation structure to resolve these conflicts. This is challenging since we may not have access to ground truth data.
- Secondly, it is essential to convey crucial lineage information about label quality to the trained end model.

The paradigm of *data programming* tackles both of these challenges by modelling multiple label sources without relying on ground truth and generating probabilistic training labels that represent the lineage of each individual label.

3 Outline of Method

3.1 Snorkel Architecture

Snorkel's workflow is designed around data programming, a fundamentally new paradigm for training machine learning models using weak supervision and is as follows:

1. *Labelling Functions*: Instead of manually labelling training data, Snorkel enables users to write labelling functions that can express various sources of weak supervision, such as patterns, heuristics, external knowledge bases, and so on. For sentiment analysis, one may use the common words occuring in particular type of texts or just use polarity expressed in sentences.

2. *Modeling Accuracies and Correlations*: Snorkel is capable of estimating the accuracies and correlations of these labelling functions, all without requiring access to ground-truth data. Snorkel achieves this by analyzing the agreements and disagreements among the labelling functions themselves. Through this process, Snorkel can produce high-quality labels, which can be used to train robust and accurate machine learning models.

3. *Training a Discriminative Model*: Snorkel produces a collection of probabilistic labels, which can be utilized to train a vast range of machine learning models, including popular deep learning models. Although the generative model that Snorkel employs is mainly a weighted combination of the user-defined labeling functions, which tend to be highly precise but possess low coverage, modern discriminative models can maintain this precision while also learning to generalize beyond the labeling functions. As a result, these models can increase coverage and robustness on unseen data, thus enhancing their performance.

3.2 Generative Model

The core task of Snorkel is modelling and integrating the noisy signals provided by a set of labelling functions. Using data programming, we model the true class label for a data point as a latent variable in a probabilistic model. In the simplest case, we model each labelling function as a noisy vote which is independent, i.e. makes errors that are uncorrelated with the other labelling functions. This defines a generative model of the votes of the labelling functions as noisy signals about the true label.
The full generative model is constructed as a factor graph. All the labelling functions $\lambda_i(x)$ are first applied to the unlabeled data points, resulting in a label matrix Λ, where $\Lambda_{i,j} = \lambda_j(x_i)$. The generative model $p_w(\Lambda, Y)$ is then encoded using three different factor types - the labelling propensity, accuracy, and pairwise correlations of labelling functions.

$$\begin{aligned}
\phi_{i,j}^{Lab} &= \mathbb{1}\{\Lambda_{i,j} \neq \emptyset\} \\
\phi_{i,j}^{Acc} &= \mathbb{1}\{\Lambda_{i,j} = y_i\} \\
\phi_{i,j}^{Corr} &= \mathbb{1}\{\Lambda_{i,j} = \Lambda_{i,k}\} \qquad (j,k) \in C
\end{aligned}$$

For a given data point x_i, we define the concatenated vector of these factors for all the labelling functions $j = 1, ..., n$ and potential correlations C as $\phi_i(\Lambda, Y)$, and the corresponding vector of parameters $w \in R^{2n+|C|}$. The model is thus defined as:

$$p_w(\Lambda, Y) = Z_w^{-1} \exp\left(\sum_{i=1}^{m} w^T \phi_i(\Lambda, y_i)\right)$$

To learn this model without access to the true labels Y, we minimize the negative log marginal likelihood given the observed label matrix Λ:

$$\hat{w} = arg\min_w - \log \sum_Y p_w(\Lambda, Y)$$

Then, the predictions, $\bar{Y} = p_w(Y|\Lambda)$, are used as probabilistic training labels.

3.3 Discriminative Model

The end goal is to train a model that generalizes beyond the information expressed in the labeling functions. A discriminative model h_θ is trained on our probabilistic labels $\bar{Y}$ by minimizing a noise-aware variant of the loss $l(h_\theta(xi), y)$, i.e. the expected loss with respect to $\bar{Y}$:

$$\hat{\theta} = arg\min_\theta \sum_{i=1}^{m} \mathbb{E}_{y\sim\bar{Y}}[l(h_\theta(x_i), y)]$$

4 Experiment Details and Main Results

4.1 Salary Prediction

To learn binary salary prediction with weak supervision, one approach is to use a classification model that can predict whether a given individual earns above or below a certain salary threshold(50k in our case). However, since there is no labeled data available to train such a model, weak supervision techniques can be used to generate labeled data from the available attributes. One way to approach this is to use rule-based or heuristic-based labeling methods, where heuristics or rules are used to infer the labels of the training data. For example, we can assume that people with higher education levels and more experience are more likely to earn higher salaries, and people in certain occupations or industries may have a higher salary range than others. We can use such assumptions to assign labels to the data based on the available attributes

To prepare the data for this approach, it is important to pre-process the data to handle missing values, categorical variables, and outliers. One common method is to use one-hot encoding to represent categorical variables, and normalization to scale the continuous variables.
Overall, the effectiveness of the weak supervision approach will depend on the quality of the heuristics or rules used to label the data and the amount and quality of the initial labeled data used to train the model. It is important to carefully validate the model's performance on a hold-out test set to ensure that the model generalizes well to new, unseen data.

The discriminative model is able to generalize beyond the noisy labels provided by the LabelModel and make good predictions on all data points, not just the ones covered by the labeling functions. This may be because the discriminative model is better able to capture the underlying patterns in the data, and is not limited by the specific assumptions or biases of the labeling functions.
By using the LabelModel to transfer the domain knowledge encoded in the LFs to the discriminative model, the approach is able to improve the accuracy of the predictions. This is likely because the LabelModel is able to capture some of the patterns in the data that are not explicitly modeled by the discriminative model, and can provide additional information that helps to improve the predictions.

Overall, this approach seems to be a good way to combine the strengths of generative and discriminative models, by using the generative model to generate noisy labels and transfer domain knowledge to the discriminative model, which can then make more accurate predictions based on that knowledge. However, the effectiveness of the approach will depend on the quality of the labeling functions and the LabelModel, as well as the specific discriminative model used to make predictions.
Classifier Logistic regression trained on a weakly supervised dataset can outperform ($accuracy$: $0.7755 \geq 0.7498$) an approach based on the LFs alone as it learns to generalize beyond the noisy heuristics we provide.
LFs: After analyzing the data we decided to come up with following :

```
@labeling_function()
def check(x):
    return 0 if int(x.hours_per_week)<40 else -1

@labeling_function()
def check2(x):
    return 0 if x.education<14 else -1

@labeling_function()
```

```
def check3(x):
    return 1 if x.capital_gain>=8000 else -1

@labeling_function()
def check4(x):
    return 0 if x.capital_loss>=3500 else -1
```

The first Lf was chosen since as hours per week increases the wage increases. Similar relations for education i.e people who have higher education i.e Phd tend to have higher wage. More gain,less loss heuristic and trying out various cutoffs we came up with the last two lfs.

4.2 Twitter Sentiment Analysis

There is a lot of text data available on popular social media sites. SWe can even get curated data by searching through hashtags. In this way we have a dataset which tries to do sentiment analysis through tweets on digital computer games. Regular sentiment models don't work well here because there is a lot of hateful and violent speech even in positive comments.Hence, weak supervision techniques can be quite effective here.

For data preparation, we fist remove the data which is marked irrelevant or neutral and also filter out the missing data. Then we try to label the remaining data without true labels using a mixture of human defined heursitics and model based technqiues. We use textblob to determine polarity of data and mark it as positive if it is greater than some value and abstain otherwise. Similar labels are done for negative sentiment with varying level of tradeoff between accuracy and coverage.
We then write two labelling functions which work by detecting keywords in texts and judging them as positive and negative. Following is a summary of the results on LFs.

	j	Polarity	Coverage	Overlaps	Conflicts	Correct	Incorrect	Emp. Acc.
polarity_positive	0	[1]	0.220027	0.089508	0.011764	7690	1774	0.812553
polarity_negative	1	[0]	0.161393	0.161393	0.003185	5759	1183	0.829588
polarity_negative_2	2	[0]	0.779973	0.330737	0.086253	20584	12965	0.613550
words_negative	3	[0]	0.161905	0.161905	0.025295	5632	1332	0.808731
words_positive	4	[1]	0.167647	0.167647	0.089903	6021	1190	0.834974

We then train a generative model using snorkel. The model is able to label the data with probablistic labels and an accuracy of around 0.70. We then use these labels to train a Recurrent neural network to learn the sentiments based on input texts. The model then achives accuracy of 0.91 on the test set using LF based labels but on true labels, it is able to beat the probablistic labels by achieving 0.73 accuracy.

4.3 Extending to continuous labels

We attempted to extend the idea of Snorkel to continuous labels. For instance, given different noise removal models, could one aggregate their produced outputs into a single 'true' noise prediction?

Our approach involved learning a latent distribution to represent the predicted labels and subsequently using this latent distribution as a probability distribution over the true label. Note that

this is identical to the idea of Snorkel. To that end, we implement a variational auto-encoder to learn the latent distribution.

Formally, given labels x, we learn $P(z|x)$, by maximizing the log likelihood of observed data. This is done by marginalizing the true latent variables by conditioning on them. VAEs are able to learn on hidden markov models however they need a prior on the latent variables which is assumed Gaussian noise. However that caanot be the case for images and so we tried to modify this.
Unfortunately, our approach did not yield the desired results. We observed that the latent representation learnt by the VAE was not directly interpretable and hence could not be used as a distribution over the true label. The VAE learnt a representation of the data that could be used for generation but the latent variable didn't 'aggregate' all labelling functions. This continued to be the case even after the prior for the latent distribution was altered.
For experiments we chose denoising images as the task. We tried it on small images of Fashion MNIST and we wrote 2 CV based LFs for denoising and sharpening. However our VAE was not able to give correct latent variables as it grew closer to decoding the input.

5 Conclusion

Snorkel provides a new paradigm for soliciting and managing weak supervision to create training data sets. Users provide higher-level supervision in the form of labeling functions that capture domain knowledge and resources, without having to carefully manage the noise and conflicts inherent in combining weak supervision sources. This significantly reduces the cost and difficulty of training powerful machine learning models while exceeding prior weak supervision methods and approaching the quality of large, hand-labeled training sets.
However extending this for regression based models is a challenging and compute heavy task. We tried to do it using VAEs however that did not work well.

6 Work split up amongst the team members

Kaustubh Ponkshe - VAEs experiment and sentiment analysis
Puranjay Datta - Salary prediction
Deshpande Hrushikesh Shriniwas - Report, PPT and Salary prediction
Sheel Shah - Report, PPT and VAEs experiment

7 References and Code

- https://arxiv.org/pdf/1605.07723.pdf
- https://arxiv.org/pdf/1711.10160.pdf
- https://arxiv.org/pdf/1911.09860.pdf

GitHub URL of codebase repo: https://github.com/sheelfshah/aml_project