MACHINE LEARNING CONCEPT:
Semi-Supervised Learning

(An in-depth tutorial for beginners)

EGWUDIKE, Oluchi
22084576

INTRODUCTION

In the ever-evolving field of machine learning, data plays a central role in training robust and accurate models (Rapl, 2023).

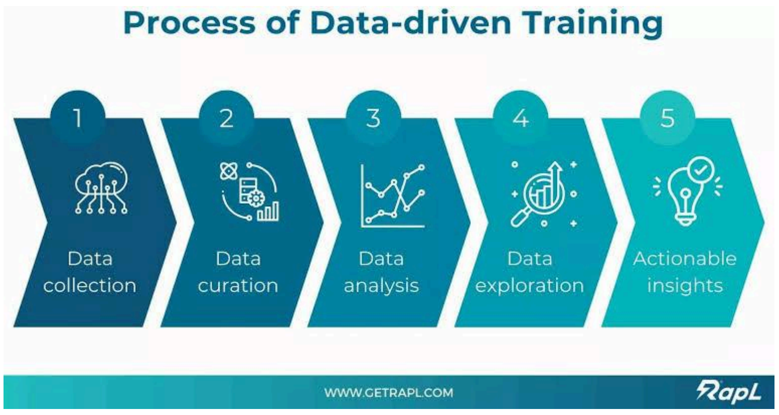

Figure 1: Showing the process involved in data driven training (source: Rapl, 2023)

However, obtaining labeled data for supervised learning can be expensive, time-consuming, and impractical in certain scenarios. Conversely, while unsupervised learning methods do not rely on labels, they often fail to provide the interpretability and accuracy needed for complex real-world tasks. Semi-supervised learning (SSL) emerges as a solution to these challenges by combining small amounts of labeled data with large volumes of unlabeled data (Databasecamp, 2023). This approach has been increasingly adopted in domains where data labeling is costly, such as medical diagnostics, natural language processing, and autonomous driving.

BACKGROUND AND CONTEXT

The need for SSL arises from the fundamental limitations of supervised and unsupervised learning as Supervised learning models require extensive labeled datasets to generalize well, which is often unfeasible in real-world applications. For instance, labeling medical images requires expert annotations, which can be prohibitively expensive and time-intensive.
On the other hand, unsupervised learning methods like clustering and dimensionality reduction do not require labels but struggle to deliver meaningful and actionable insights.

SSL bridges this gap by using the distribution of unlabeled data to guide the learning process. For example, if labeled data indicates that certain features correspond to a specific class, SSL algorithms can use the distribution of these features in the unlabeled dataset to refine the model. This paradigm allows for more efficient use of available data, enhancing model accuracy and reducing reliance on labeling.

Several methodologies have been developed to implement SSL effectively (Abdulrazzaq et al., 2024). These approaches leverage the relationships between labeled and unlabeled data in unique ways:

1. **Self-Training**: Self-training is one of the simplest SSL techniques. The process begins with training a model on the available labeled data. This initial model is then used to predict labels for the unlabeled data, generating pseudo-labels. These pseudo-labeled samples are added to the training set, and the model is retrained iteratively. While self-training is straightforward, its success hinges on the accuracy of the initial model. If the initial model generates incorrect pseudo-labels, these errors can propagate, leading to a phenomenon known as model collapse.
2. **Co-Training**: Co-training takes the self-training approach a step further by using multiple models trained on different subsets of features. For example, in an email spam detection task, one model might focus on the email content while another uses metadata such as sender information. Each model generates pseudo-labels for the unlabeled data, which are then used to retrain the other model. This cross-training reduces the risk of error accumulation, as the models rely on independent views of the data.
3. **Generative Models**: Generative models estimate the joint probability distribution of the features and the labels, allowing them to make predictions for unlabeled data. Naïve Bayes and Gaussian Mixture Models (GMMs) are popular generative approaches in SSL. These methods use algorithms like Expectation-Maximization (EM) to iteratively refine the model parameters and predict soft labels for the unlabeled data. While generative models are computationally efficient for small datasets, they may struggle with scalability and complex feature dependencies.
4. **Graph-Based Methods**: In graph-based SSL, data points are represented as nodes in a graph, with edges connecting similar nodes. Label propagation is a common technique in this category, where labels from labeled nodes are iteratively propagated to their neighbors. The graph structure captures the relationships between data points, making this approach particularly effective for tasks like community detection in social networks.

APPLICATIONS

The versatility of SSL makes it applicable across a wide range of domains. Below are some prominent use cases (Wilson, 2019):

- **Natural Language Processing (NLP):** Semi-supervised learning is widely used in tasks like sentiment analysis and machine translation. For example, in low-resource languages, SSL enables models to leverage unlabeled text data to improve translation quality.
- **Medical Imaging**: Diagnosing diseases from medical images often requires annotated datasets created by medical experts. SSL helps overcome this limitation by using

unlabeled images to improve the accuracy of models for tasks such as tumor detection and organ segmentation.
- **Autonomous Driving**: Self-driving cars rely on machine learning models to recognize objects, predict trajectories, and make decisions in real time. SSL enables these models to utilize vast amounts of unlabeled driving data, reducing the dependency on expensive labeled datasets.
- **Fraud Detection**: In financial systems, labeled datasets for fraud detection are limited and often imbalanced. SSL algorithms can use transaction patterns from unlabeled data to improve detection rates.

CHALLENGES AND LIMITATIONS

While SSL offers numerous advantages, it also comes with its own set of challenges:

- **Noisy Pseudo-Labels**: In techniques like self-training, incorrect pseudo-labels can lead to error propagation, diminishing model performance.
- **Overfitting to Unlabeled Data**: Balancing the contributions of labeled and unlabeled data is critical. Excessive reliance on unlabeled data can lead to overfitting, where the model fails to generalize to new data.
- **Hyperparameter Sensitivity**: SSL methods often require careful tuning of hyperparameters such as confidence thresholds, regularization weights, and the number of iterations. Poor choices can significantly impact model performance.
- **Scalability**: Some approaches, such as graph-based methods, may struggle to scale with large datasets due to computational complexity.

FUTURE DIRECTIONS AND CONCLUSION

As the availability of unlabeled data continues to grow, SSL is poised to play a crucial role in the future of machine learning. Advancements in techniques like self-supervised learning, which generates its own labels from data, and transfer learning, which adapts pre-trained models to new tasks, are complementing the capabilities of SSL (IEEE, 2024; Chen, 2024; MDPI, 2023).

In conclusion, semi-supervised learning is a powerful paradigm that addresses the limitations of traditional learning approaches. By leveraging both labeled and unlabeled data, it opens new possibilities for solving complex problems in various domains. As research continues to enhance SSL methodologies, its adoption in real-world applications is expected to expand, making it an indispensable tool in the machine learning toolkit.

PRACTICAL DEMONSTRATION USING SSL TECHNIQUES

To further consolidate the knowledge of Semi Supervised Learning techniques, a practical machine learning exercise will be carried out to perform a sentiment analysis task on the IMDB dataset simulating a scenario where only a small portion of labeled data is available. Students should try to recreate it to fully understand the application and concepts.

The purpose of this implementation is to demonstrate the self-training approach in semi-supervised learning (SSL). Self-training is one of the simplest SSL techniques, which iteratively refines a machine learning model by leveraging both labeled and unlabeled data.

In real-world scenarios, obtaining labeled data is often expensive and time-consuming, whereas unlabeled data is abundant. Self-training bridges this gap by allowing the model to generate pseudo-labels for the unlabeled data and uses these pseudo-labels to improve its predictive performance.

The implementation begins with loading and preprocessing the IMDB movie review dataset from the NLTK library.

1. To begin, we will import the libraries to be used for this task

Code snippet:

```
import numpy as np
import pandas as pd
from sklearn.model_selection import train_test_split
from sklearn.feature_extraction.text import CountVectorizer
from sklearn.linear_model import LogisticRegression
from sklearn.metrics import accuracy_score
```

Next, is to Load the data set into a variable:

```
#Loading the IMDB dataset from NLTK
from nltk.corpus import movie_reviews
import nltk
nltk.download('movie_reviews')

# Preparing the dataset
documents = [(list(movie_reviews.words(fileid)), category)
          for category in movie_reviews.categories()
          for fileid in movie_reviews.fileids(category)]

random.shuffle(documents
```

The dataset is then encoded and split up into reviews and labels, labelled and unlabelled, validating and testing (check the attached jupyter notebook to understand the code write up)

2. Initial Model Training
A logistic regression classifier is trained on the small labeled dataset using a bag-of-words text representation. We will Train an initial model capable of making predictions on unseen data The initial model achieves moderate accuracy on the validation set, serving as a baseline for further improvements.

Code snippet:

```
#Creating a vectorizer and logistic regression model
vectorizer = CountVectorizer(stop_words='english', max_features=5000)
classifier = LogisticRegression(solver='liblinear', random_state=42)
```

3. Self-Training Loop
The self-training process iteratively refines the model by adding pseudo-labeled high-confidence samples from the unlabeled dataset. Predict pseudo-labels for the unlabeled data using the current model, Select predictions with high confidence (e.g., probabilities > 0.8), Add these pseudo-labeled samples to the training set and lastly we retrain the model with the expanded training set.

Code snippet:

```
# Self-Training Parameters
confidence_threshold = 0.8
iterations = 5
validation_accuracies = []

# Self-Training Loop
for i in range(iterations):
    print(f"\nIteration {i+1}:")
```

4. Evaluation
The final model is evaluated on a hold-out test set to measure its predictive performance. The final model achieves higher test accuracy compared to the initial model, demonstrating the effectiveness of self-training.
Code snippet:

```
# Evaluate the model on the validation set
    y_val_pred = model.predict(X_val)
    val_accuracy = accuracy_score(y_val, y_val_pred)
```

5. Visualization
A plot of validation accuracy across iterations provides insights into the model's learning progress and visualize how self-training impacts model performance over time.

Code snippet:

```
# Plotting the validation accuracies
plt.figure(figsize=(10, 6))
plt.plot(range(1, iterations + 1), validation_accuracies, marker='o', linestyle='--', color='b')
plt.title("Validation Accuracy Across Iterations")
plt.xlabel("Iteration")
plt.ylabel("Validation Accuracy")
plt.xticks(range(1, iterations + 1))
plt.grid()
plt.show()
```

At the end of the code implementation and processing a machine learning report will be created for an in-depth explanation on the outcome of this SSL process.

CONCLUSION

The self-training approach exemplifies the power of semi-supervised learning in scenarios where labeled data is scarce. By combining a small labeled dataset with a large pool of unlabeled data, this method enables robust model training while minimizing labeling costs. The implementation demonstrates how a simple iterative process can yield significant performance gains, making self-training a valuable tool in the machine learning toolkit.

REFERENCES

Abdulrazzaq, M. M., Ramaha, N. T. A., Hameed, A. A., Salman, M., Yon, D. K., Fitriyani, N. L., Syafrudin, M., & Lee, S. W. (2024). Consequential advancements of Self-Supervised Learning (SSL) in deep learning contexts. Mathematics, 12(5), 758. https://doi.org/10.3390/math12050758

BP. (2024, September 27). Unsupervised Learning: Types and challenges | BotPenguin. https://botpenguin.com/glossary/unsupervised-learning

Lang, N. (2023, November 11). What is Semi-Supervised Learning? | Data Basecamp. Data Basecamp. https://databasecamp.de/en/ml/semi-supervised-learning-en

Wilson, A. (2019, January 9). What is Semi-Supervised Learning? Try Machine Learning. https://trymachinelearning.com/what-is-semi-supervised-learning/