STAT8056 Assignment 2, due 4/15/2023 in class

You may choose one of the three problems, each worth 10 points. If you attempt an additional problem, a maximum of 5 bonus points will be awarded. If you choose Problems 1 or 2, you will need GPU for training/inference, and you may use Google Colab for a T4 GPU free of charge.

Problem 1 (Sentiment Analysis with BERT-Style Transformers, 10 points) This problem focuses on constructing sentiment classifiers using the IMDB movie review dataset [3]. The goal is to predict the sentiment of each review as either "Positive" or "Negative". The dataset comprises 50,000 reviews, of which half is for training and while the other is for testing, and an equal division of positive and negative reviews in both sets.

Your task is to perform sentiment classification using the training set and evaluate the performance of the test set. Consider the following steps:

(a) Implement sentiment classification using traditional methods such as RNNs and LSTMs.

(b) (Knowledge Transfer) Fine-tune a BERT-style transformer, specifically DistilBERT [4] (see below), for sentiment classification. Fine-tuning can vary depending on computational budgets, and here are two conventional approaches [1]:

  (i) **Feature-based approach:** Extract embeddings from the pre-trained transformer and use these as features to build a classifier. Possible classifier choices include logistic regression, random forests, gradient boosting, and fully connected neural networks.

  (ii) **Full fine-tuning:** Append additional dense layers to the transformer's embeddings and optimize all model weights. Unlike in the feature-based approach, here, the weights of the pre-trained transformer are not fixed and are fine-tuned along with the appended layers.

(c) Compare classification results from the models in (a) and (b). Discuss their advantages and disadvantages.

Details about data, implementation codes and platforms, and models:

- IMDB movie review dataset: https://huggingface.co/datasets/imdb
- RNN in PyTorch: https://pytorch.org/docs/stable/generated/torch.nn.RNN.html
- LSTM in PyTorch: https://pytorch.org/docs/stable/generated/torch.nn.LSTM.html
- DistilBERT: https://huggingface.co/distilbert-base-uncased
- Conventional ways of fine-tuning by Sebastian Raschka: https://sebastianraschka.com/blog/2023/pytorch-faster.html
- Sentiment analysis with DistilBERT and some acceleration tricks: https://magazine.sebastianraschka.com/p/finetuning-large-language-models

[1] For advanced fine-tuning techniques suitable for LLMs, refer to parameter-efficient fine-tuning (PEFT)

Problem 2 (Tabular Data Generation with Diffusion Models, 10 points) In this problem, your task is to train a diffusion model to generate synthetic tabular data using the Adult dataset [1]. The synthetic data will closely mirror the distribution of the original data, making it suitable for various downstream learning tasks. You will answer the following questions:

(a) Split the Adult dataset into training, validation, and testing sets, roughly each of equal size.

(b) Train a Tabular Denoising Diffusion Probabilistic Model (TDDPM, [2]) on the training set.

(c) Generate synthetic data and compare its distribution with that of the test data. Evaluation metrics may include pair-wise correlations and some distributional distances of your choice (You may consider the Kolmogorov-Smirnov distance, the 1- or 2-Wasserstein distance, or other reasonable distributional distances).

(d) Use the generated synthetic data to build a predictive model trained on the synthetic data. Compare its performance with a model trained on the original training data. Use the original validation set to determine the model's tuning parameters for preventing overfitting and the original testing set for final evaluation.

Details about data, implementation codes and platforms, and models:

- Adult income dataset: https://archive.ics.uci.edu/dataset/2/adult
- GitHub repo: TabDDPM — Modelling Tabular Data with Diffusion Models: https://github.com/yandex-research/tab-ddpm
- Evaluation metrics: https://arxiv.org/pdf/2310.09656v1.pdf (Section E.3)
- Wasserstein distance: [Definition | Implementation]

Problem 3 (Recommender Systems, 10 points) This problem examines the Movielens dataset, available at http://grouplens.org/datasets/movielens/. This dataset, collected via the MovieLens website (movielens.umn.edu) between September 19th, 1997, and April 22nd, 1998, has undergone preprocessing to enhance data quality. Here, you will focus on the 100K MovieLens dataset, including 100,000 anonymous ratings on a five-star scale from 1,000 users across 1,700 movies. It features four user-related categorical covariates—gender, age, occupation, and zip code (first digit only)—and one content-related continuous covariate, genres, reparametrized into 19 binary covariates representing movie genres.

The tasks are as follows:

(a) Develop a model to predict user preference scores for all movies using a method of your choice, such as deep learning. Compare your approach with the Singular Value Decomposition (SVD) method. Utilize the five pairs of training and testing datasets (u1.base, u1.test, $\cdots$, u5.base, u5.test) provided for five-fold cross-validation. Compute the Root Mean Square Error (RMSE) based on this cross-validation approach.

(b) Investigate the pattern of missing ratings in the dataset. Provide evidence to argue whether the missing ratings occur at random or not. (Hint: Use suitable plots to support your argument.)

(c) If the objective is to develop a recommender system that outperforms conventional SVD methods in terms of prediction accuracy, refine your method to account for the observed pattern of missing ratings. Evaluate whether this refinement improves prediction performance compared to the original method.

Resources you might find useful for this problem:

- Movielens dataset: http://grouplens.org/datasets/movielens/
- R-package: gsrs: A Group-Specific Recommendation System: https://cran.r-project.org/web/packages/gsrs/gsrs.pdf
- Matlab code available at https://sites.google.com/site/xuanbigts/software

References

[1] R. Kohavi et al. Scaling up the accuracy of naive-bayes classifiers: A decision-tree hybrid. In *Kdd*, volume 96, pages 202–207, 1996.

[2] A. Kotelnikov, D. Baranchuk, I. Rubachev, and A. Babenko. Tabddpm: Modelling tabular data with diffusion models, 2022.

[3] A. L. Maas, R. E. Daly, P. T. Pham, D. Huang, A. Y. Ng, and C. Potts. Learning word vectors for sentiment analysis. In *Proceedings of the 49th Annual Meeting of the Association for Computational Linguistics: Human Language Technologies*, pages 142–150, Portland, Oregon, USA, June 2011. Association for Computational Linguistics.

[4] V. Sanh, L. Debut, J. Chaumond, and T. Wolf. Distilbert, a distilled version of bert: smaller, faster, cheaper and lighter. *arXiv preprint arXiv:1910.01108*, 2019.