ELV832: Assignment, Part 2

Sumeet Agarwal

April 20, 2025

Weightage: 21% of course grade
Deadline: 29th April 2025

The basic procedure to follow is pretty similar to Part 1. We need to generate some toy data from a teacher model, and then train a student model for varying levels of sample complexity, trying to study how the generalisation error and nature of learning of the student model are impacted by the level of sample complexity. However, here we will be using a more sophisticated transformer architecture, as in the experiments of Cui et al. (2024), in particular Section 5 of that paper.

1 Teacher model

The model is being presented here using the notation also written down in class. The paper has essentially the same model, but uses different notation.

The below is the process for generating a *single data point* from the teacher model. This process can be repeated N times to generate a data set of size N. We will want to set $N = 2.2D$, just as in Part 1.

We will set the number of words per sentence $L = 2$. The dimensionality of the embedding vectors can initially be set to $D = 100$; later on you should try to increase this as much as possible, as we are interested in studying the behaviour in the high-dimensional limit. So we first generate the input (isolated) word embedding vectors:

$$\mathbf{x}_1, \mathbf{x}_2 \sim \mathcal{N}(\mathbf{0}, \sigma^2 \mathbb{I}_D),$$

where we set $\sigma^2 = 0.25$.

Now, the idea is to generate the *target* attention weights as a mixture of a semantic component and a positional component:

$$\forall l, m \in \{1, ..., L\}:$$
$$a^*_{lm} = (1-\omega)\text{softmax}\left(\frac{\mathbf{x}_l^{\mathrm{T}} W_Q^{*\mathrm{T}} W_Q^* \mathbf{x}_m}{\sqrt{D_K}}\right) + \omega f(l, m).$$

So the semantic part is the standard scaled dot-product self-attention, with shared *target* key and query weights denoted by W_Q^*. On the other hand, the positional part is specified by the function $f(l, m)$, which is purely a function of the positions of the 2 tokens being considered, and not of their embedding vectors.

To complete the specification of the target attention weights to be generated by the teacher model, we make the following specific parameter choices initially (but try to write your code so that you can easily vary any of these for later experiments):

$$\begin{aligned}
\omega &= 0.3 \\
D_K &= 1 \\
W_Q^{*\mathrm{T}} &\sim \mathcal{N}(\mathbf{0}, \mathbb{I}_D) \\
f(l, m) &= \begin{cases} 0.6; l = m \\ 0.4; \text{otherwise} \end{cases}
\end{aligned}$$

Having fully specified how to generate the target attention weights, we can now use these to generate the actual *target output vectors* for each token:

$$\forall l \in \{1, ..., L\}:$$
$$\mathbf{t}_l = \sum_{m=1}^{L} a^*_{lm} \mathbf{x}_m.$$

So this completes the specification for how to generate one data point. In general, we could denote the n^{th} data point by the tuple $\{X^n, T^n\}$, where

$$X^n = \begin{pmatrix} \mathbf{x}_1^{n\mathrm{T}} \\ . \\ . \\ . \\ \mathbf{x}_L^{n\mathrm{T}} \end{pmatrix}; \; T^n = \begin{pmatrix} \mathbf{t}_1^{n\mathrm{T}} \\ . \\ . \\ . \\ \mathbf{t}_L^{n\mathrm{T}} \end{pmatrix}.$$

Hence, a *data set* will be of the form $\{X^n, T^n\}_{n=1}^N$. Your code should be able to generate and appropriately store such a data set, for a given set of hyperparameter choices as indicated above.

2 Student model training and analyses

2.1 Main attention-based student model

This will be a simple transformer model with a single attention layer (with a single head) and nothing else. To allow the model to learn positional relationships, we need to use positional embeddings as usual. Since we have $L = 2$, there are only 2 positions to be represented within each sentence, and hence we can simply set the positional encodings as

$$\mathbf{r}_1 = -\mathbf{r}_2 = \mathbf{1}_D.$$

Hence the student model-computed attention weights for a given input sentence X will be

$$\begin{aligned} &\forall l, m \in \{1, ..., L\} : \\ &a_{lm}(X; W_Q) = \text{softmax}\left(\frac{(\mathbf{x}_l + \mathbf{r}_l)^{\mathrm{T}} W_Q^{\mathrm{T}} W_Q (\mathbf{x}_m + \mathbf{r}_m)}{\sqrt{D_K}}\right), \end{aligned}$$

where W_Q is the $D_K \times D$ matrix of learnable key/query parameters. We will set $D_K = 1$ as for the teacher model. Then, the model outputs for each token in a sentence X are given by

$$\begin{aligned} &\forall l \in \{1, ..., L\} : \\ &\mathbf{y}_l(X; W_Q) = \sum_{m=1}^{L} a_{lm}(X; W_Q)(\mathbf{x}_m + \mathbf{r}_m). \end{aligned}$$

Thus, for a data set of the form $\{X^n, T^n\}_{n=1}^N$ as generated from the teacher model above, we can define the regularised sum-of-squares error for the student model as

$$E(W_Q) = \sum_{n=1}^{N} \frac{1}{2D} \sum_{l=1}^{L} ||\mathbf{t}_l^n - \mathbf{y}_l(X^n; W_Q)||^2 + \frac{\lambda}{2}||W_Q||^2.$$

Here the λ hyperparameter should be tuned via cross-validation as usual. You will need to train your student models by minimising the above error using batch gradient descent in PyTorch. Two different initialisations will be used for each model: $W_Q = W_Q^*$ (semantic) and $W_Q = \mathbf{r}_1^{\mathrm{T}}$ (positional).

2.2 Dense linear baseline model

For comparison, we will also train via the same kind of error function the following baseline linear model:

$$\forall l \in \{1, ..., L\} :$$
$$\hat{\mathbf{y}}_l(X; W) = \sum_{m=1}^{L} w_{lm}\mathbf{x}_m,$$

where $W = \begin{pmatrix} w_{11} & w_{12} & ... & w_{1L} \\ w_{21} & ... & & \\ ... & & & \\ w_{L1} & ... & & w_{LL} \end{pmatrix}$ is the $L \times L$ learnable parameter matrix.

3 Experiments to be done

Your primary goal should be to reproduce the computational experiments depicted in Figures 2 and 3 of Cui et al. (2024), using the above setup which essentially matches that specified in the paper. The key independent variable whose effects you are studying is the sample complexity $\alpha = N/D$, where N refers to the size of the training set. A range of values of $\alpha \in (0, 2]$ has to be tried in most of the experiments, just as in Part 1 of the Assignment. Apart from looking at training loss, you should also do versions of the same plots which use the *test error* instead (since in every data set from the teacher model you have generated 10% extra data which is to be used only for testing).

A couple of additional summary statistics which help to interpret what the attention-based model has learnt are the following:

$$m = \frac{W_Q\mathbf{r}_1}{D};$$
$$\theta = \frac{\sigma^2 W_Q W_Q^{*\mathrm{T}}}{D}.$$

These can be seen as capturing the overlap of the finally learnt weights with, respectively, the positional and semantic initialisations as mentioned above. So you should try calculating and plotting these in the same way as in Figures 2 and 3.

In terms of variations in the data set or teacher model, the key hyperparameters that should be varied there are D and ω, at least over the ranges indicated in

Figures 2 and 3.

You should submit (via Moodle) all your code for all of the above models and experiments, clearly organised and commented so as to be understandable by others. All your results/plots should be organised into a report, which should have all the details of the variations you have tried and the outcomes obtained, as well as how you think they should be interpreted. In particular, you should try to explain any differences or novelty compared to the results of Figures 2 and 3.