DREAM Challenge 2022
Predicting gene expression using millions of random promoter sequences
by
Team BUGF

Abstract

An attempt was made to predict gene expression levels from a set of random promoter sequences as part of a 2022 DREAM Challenge (de Boer et al. 2020). A transformer encoder model was used with shared weights to predict the expression bin classes, as opposed to treating the problem as a regression problem. An interesting data augmentation strategy was used to reduce the effects of overfitting, whereby the prediction confidence on 'fake' promoter sequences was considered alongside a more standard loss function.

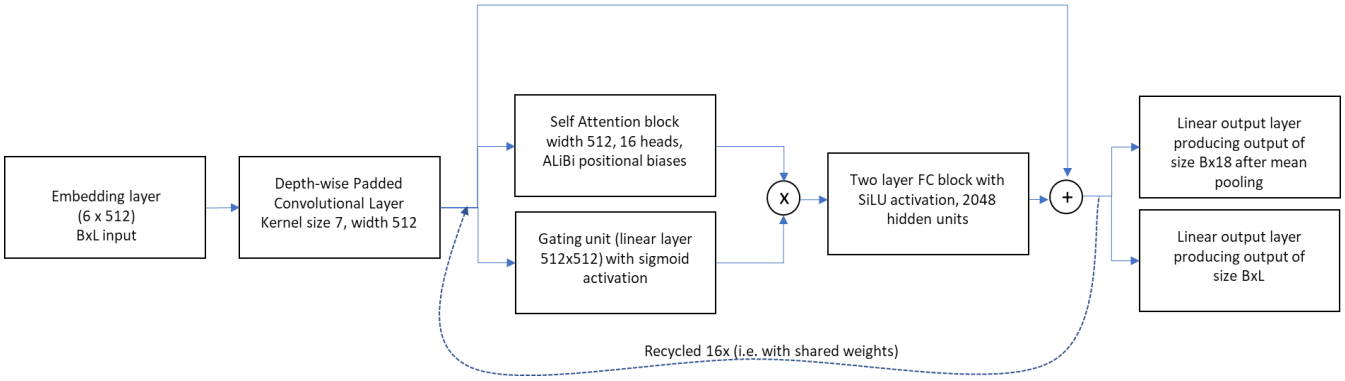
1. Description of data usage

A single random split of the training set was made at the very start so that progress could be tracked:
6357910 training_set.txt
155123 validation_set.txt
226225 holdout_set.txt

The holdout set was ultimately not used beyond the first leaderboard week as the validation set results proved to be accurate enough for model selection without it.

2. Description of the model

The final architecture (see diagram below) settled upon was a fairly standard self-attention encoder model (Vaswani et al. 2017) with a single depthwise convolution at the start with kernel size 7 (see Chu et al. 2021) and ALiBi relative positional biases (Ofir et al. 2021). Model size was embedding dimension 512, 16 heads and 16 layers, however all of the layers shared weights to reduce overfitting. Dropout of 0.1 was used within both the self-attention block and the feedforward block of the shared encoder, and gating similar to that used in AlphaFold2 (Jumper et al. 2021) was used to modulate the output of the self-attention function. The task was actually treated as a classification problem rather than a regression problem, with the prediction targets being the bin label (0-17) after average pooling over the final hidden encoder state. To produce continuous single values, the probability vector (after softmax normalization) was multiplied by a vector of constants [0,1,2 … 16,17] and summed. An auxiliary output is also produced which was a single probability per input token aimed at predicting the locations of random mutations applied to the input sequence during training, this is used to condition the model i.e. to try to prevent early overfitting.



3. Training procedure

Training was carried out in minibatches of 32 per TPU core (256 in total), using PyTorch 1.12 (with XLA) on Google cloud v2-8 VM instances. Two loss functions were used. The primary loss function was multi-label focal loss (Lin et al. 2017) with 18 label classes and γ set to 2. An auxiliary loss was used to reduce overfitting of the model, and this was binary cross-entropy loss trained to predict where random mutations had been made to the target sequence. A mutation rate of 15% was used to generate these inputs. Note that the expression predictions were made with the original unchanged sequence, requiring inference to be carried out twice per sample. These two losses were added together for each minibatch. Weight optimization was performed using the RAdam minimizer (Liu et al. 2019), with a maximum learning rate of $5x10^{-4}$. Early-stopped training was effectively carried out by only saving the model weights which produced a minimum of validation loss. For validation loss the function 1 – r was used,where r is the Pearson correlation coefficient between predicted and real expression values in each validation minibatch.

As some of the observed labels were given as non-integers, this uncertainty was handled during training by adding a uniform random number in the range [0,1) to each label before truncating to an integer label. This random sampling effectively takes into account the original label uncertainty during training.

4. Other important features

Float32 representations were used throughout.

5. Contributions and Acknowledgement

5.1 Contributions

Name	Affiliation	Email
David T. Jones	University College London	d.t.jones@ucl.ac.uk

5.2 Acknowledgement

I thank members of the UCL Bioinformatics Group for useful discussion of possible approaches to the challenge.

6. References

de Boer, C. G. et al. (2020). Nature biotechnology, 38(1), 56-65.
Vaswani et al. (2017) arXiv:1706.03762
Chu et al. (2021) arXiv:2102.10882

Ofir et al. (2021) arXiv:2108.12409
Jumper et al. (2021) Nature 596, 583–589.
Lin et al. (2017) arXiv:1708.02002
Liu et al. (2019) arXiv:1908.03265