Agastya Gupta
Ahsan Azim

Convolutional Neural Networks for DNA Sequence Classification

Aim / Hypothesis

The aim of our project remains identical to that introduced previously – i.e. exploring the possibilities of Convolutional Neural Networks (CNNs) for DNA Sequence Classification. We continue to move forward with our experimentation into the efficacy of CNNs this task, and have received encouraging results on that front in the past week. Indeed, our prediction that CNNs may prove effective at such classification has already been vindicated – our architecture in some cases surpasses, and other cases matches those of both non-CNN and existing CNN-based efforts at such DNA Sequence classification problems.

Thus the first part of our initial hypothesis – i.e. proving the efficacy of our algorithm on smaller canonical datasets – has been cleared. We shall now move on to extending our research to relatively untouched datasets, as well as conducting further examination into the robustness of our current architecture.

Computational approaches developed

As indicated by our project's aim, development efforts have focused on Convolutional Neural Networks. This has involved tasks such as acquiring relevant theoretical domain knowledge required to understand CNNs, and conducting experimentation into how to apply such knowledge. As our goal was to both implement our own CNN, *and* use external libraries to conduct novel research, we were faced with the opposite tasks of fully understanding a CNN's internal working, as well as gaining "street smarts" relevant to knowing what works in the wild. This has allowed us to side-by-side develop a CNN of our own, and also use existing implementations with the goal of obtaining research-worthy accuracies and findings.

When it comes to the *how* of developing a CNN and understanding how to use such structures well for real-world analysis, we perused papers by luminaries in the field such as LeCunn (the pioneer of CNNs) and Andrew Ng. We also read widely on prior successful CNN classification jobs – most of which reside in the field of Computer Vision. We thus had to transfer a computer vision approach to handling sequence data, and were led to literature on using CNNs for text classification. This proved key to bridging the divide, as treating the classification of sequence data as another form of text classification allowed us to use CNNs in this domain. We took measures to retain the biological integrity of the data, since clearly human sentences are structured and interpreted far differently from DNA sequences. Some subtle matrix operations – e.g. one-hot vectors using a dictionary of all possible k-mer DNA sequence permutations – formed the core of such a synthetic approach.

Till yesterday we had been using a CNN architecture with two 2D Convolution layers, a 2D Max Pooling layer, Dropout layers, a Fully Connected Layer, and obviously an output layer. We also chose softmax as our loss function. Such an implementation matched the accuracy of our paper of focus (i.e. *Nguyen et al.*).

Taking into consideration the relatively smaller size of our data, we tried a test-run with the identical architecture, barring that this time there was only one Convolutional Layer,

since we feared that the presence of two layers was causing overfitting. Our hypothesis (we think!) proved correct, as this setup's maximum accuracy surpassed that of the paper's.

Data used

We focused on two datasets for our current experimentation – a dataset on Splice junctions in Primates DNA sequences, and a dataset on *E.Coli* Promoter DNA sequences. The two datasets are canonical when it comes to the applicability of Machine Learning methods to Biological Classification problems, and both contain DNA sequences of length ~ 57. The Splice junction dataset was of far greater size – 3000+ sequences as opposed to only 100 or so in the Promoter dataset – and thus more amenable to our experiments, since Deep Learning tools operate best on large amounts of data. It also proved favourable because Splice junction determination presents a multi-class classification problem, as opposed to the simpler binary problem in Promoter sequences. A note on the biological nature of the problems here – Splice junction determination refers to the problem of determining whether a given DNA sequence is an Intron-Exon boundary (i.e. "Acceptor"), an Exon-Intron boundary (i.e. "Donor"), or neither. The Promoter dataset, on the other hand, presents the simpler labels of "+" and "-" for each DNA sequence, with said labels respectively referring to the presence and absence of Promoters in the sequence.

Current results

Although we mentioned highlights of our results in the "Computational Approaches" section, we shall expand upon the situation here. The table below presents the most noteworthy data:

Architecture	Max accuracy % on Splice Junction dataset	
Nguyen et al.	96.72	
Ahsan and Agastya	**96.86** (1 Conv layer)	96.23 (2 Conv layers)

Although we are still investigating possible reasons as to just why our single-convolutional layer implementation proved most successful, we think our theory relating to a lack of overfitting is the most likely reason. There is, though, a bit more subtlety to the question then what has just been mentioned, and we shall continue investigating optimal parameters and hyperparameters. One key part of this search was determining an optimal batch size. This is of great importance in such architectures, and after quite a bit of experimentation, we found 31-32 to be optimal. Any higher or lower usually engendered worse accuracies. We tuned other parameters as well – a full description of this process and its results (trend lines, etc) will be included in our final paper.

We have a record of outputs of various noteworthy test-runs in the readme of our github repository (https://github.com/ahsanazim/cnn), while more detailed records are kept locally. Ultimately we will compile all our collected data and present several visual analyses of various parameters, data types, architectures, and their effect on the accuracy of our classification.

To do list[1]

As we have cleared the initial goal of proving that we can successfully classify canonical datasets, we can now move on to other parts of our hypothesis. We plan to do all of the following in the coming days:

- Continue tuning our architecture to achieve better accuracy; also examine its scalability and efficacy with larger and more varied data.
- Finish off our personal CNN implementation. This has proved rather time-consuming due to the sheer number of specific details that need to be taken care of. We are, though, on the right track.
- Develop a user-friendly web application to allow anyone, anywhere, to classify DNA sequences that we have developed optimal classification architectures for.
- Extend our architecture to novel datasets that have perhaps never been tackled before.
- Investigate Probabilistic Graphical Models in general, and the Markov Model family in particular, as a possible technique for optimizing our CNN.
- Try and articulate, in text, why we feel CNNs are so effective at this task. This will particularly enjoyable, since the reasons stretch across Deep Learning, Natural Language Processing, and Biological domains. We have researched the matter previously, but have not yet prepared a concrete document on why CNNs are proving effective right now.

Expected results

As our project has moved faster than one might expect, and we have met the basic aims in our initial goal set, we can now move to the even more novel problems mentioned in the todo list above. Setting out expected results for the kind of activities above may be hard, yet we have a few results we hope shall come to pass.

We would like to at the very least further investigate what sort of calibration for our CNN achieves maximum accuracy. Perhaps our current result is indeed said maximum; either way, it behoves us to continue investigating possible improvements. At worst we shall gain a better understanding of CNNs and the determinants of their accuracy.

We would also like to obtain positive results in our forthcoming investigation into the possible usefulness of Probabilistic Graphical Models as an optimization tool for our classification.

Most interesting of all, though, would be success in applying our architecture to novel datasets. This would be a good examination of its robustness, and perhaps faults revealed during said process would help improve our basic CNN implementation.

[1] Note that we felt that a literal list was indeed the best way to present the above information – hopefully that is not improper for such a paper?

References

Nguyen, N.G., Tran, V.A., Ngo, D.L., Phan, D., Lumbanraja, F.R., Faisal, M.R., Abapihi, B., Kubo, M. and Satou, K. (2016) DNA Sequence Classification by Convolutional Neural Network. J. Biomedical Science and Engineering, 9, 280-286.
http://dx.doi.org/10.4236/jbise.2016.95021

S. Min, B. Lee, and S. Yoon. Deep learning in bioinformatics. Briefings in Bioinformatics, in press, 2016.
https://arxiv.org/pdf/1603.06430.pdf

Haoyang Zeng, Matthew D. Edwards, Ge Liu, David K. Gifford; Convolutional neural network architectures for predicting DNA–protein binding. *Bioinformatics* 2016; 32 (12): i121-i127. doi: 10.1093/bioinformatics/btw255
https://academic.oup.com/bioinformatics/article/32/12/i121/2240609/Convolutional-neural-network-architectures-for

Ke Chen Lukasz, A. Kurgan; Neural Networks in Bioinformatics, pp 565-583
http://link.springer.com/referenceworkentry/10.1007%2F978-3-540-92910-9_18

Stanford CS 231n Convolutional Neural Networks class
http://cs231n.github.io/

Notes from Andrew Ng's Coursera class on Machine Learning
http://www.holehouse.org/mlclass/

Convolutional Neural Networks tutorial with Keras
https://elitedatascience.com/keras-tutorial-deep-learning-in-python