Predicting the neurobiological origins of childhood psychopathology

Ben Kotopka
bkotopka@stanford.edu

Archa Jain
archa@stanford.edu

David Liu
dzliu@stanford.edu

Ali Sharafat
sharafat@stanford.edu

Owen Phillips
owenp@stanford.edu

Abstract

Many human traits, such as height, are influenced by thousands of different genetic loci. While studies collecting genome-wide information from thousands of subjects are now feasible, no current modeling strategy can effectively learn phenotype-genotype relationships for these traits. Using a dataset containing genomic and psychiatric information for over 8000 subjects, we developed deep learning approaches to study the relationship between an individual's genetics and psychiatric status. We demonstrated that complex psychiatric datasets can be reduced to information-rich encodings using autoencoders, and had some success predicting individual labels which demonstrate that genetic data provides some signal for phenotypic inference. Our results demonstrate that machine learning approaches are a promising avenue for tackling this extremely difficult problem and we expect to see better results if these techniques were applied to a larger dataset, offsetting the need for aggressive dimensionality reduction.

1 Introduction

Despite the large amount of human genetic data collected in recent years, little is known about how an individual's genetics influences their phenotype. The development of "SNP chips", which determine which alleles are present at hundreds of thousands of common single nucleotide polymorphism (SNP) sites, enabled genome-wide association studies (GWASes) to identify SNP variants which are statistically associated with a trait of interest. However, only a small percentage of the natural variability in most complex traits can be explained by existing models of GWAS-identified SNPs [1, 2].

Understanding the genetic origins of psychiatric disorders is especially difficult. While a number of large GWAS studies have been done in this area, few highly significant alleles have been found and attempts to build predictive models based on the GWAS results have largely failed (see Related Work). The problem is compounded by the difficulty of objectively assessing the mental state of a large number of subjects; comprehensive psychiatric exams can generate hundreds of response variables. In order to make progress in this area, a method that can tolerate large datasets and learn complex relationships among variables is needed.

Recent breakthroughs in deep learning methods have revolutionized fields such as computer vision and natural language processing. Deep learning is a type of machine learning algorithm that leverages very large datasets to find hidden structures within them and makes accurate predictions. In this paper we apply a variety of deep learning methods on human genetic data to predict psychiatric disorders. Despite the difficulty of the learning task, our results show that there is a weakly observable correlation between human genetic data and some psychiatric disorders.

2 Related Work

The current state of the art in phenotypic inference from SNP data is the genome-wide polygenic score (sometimes called a GPS), which attempts to model a trait based on a set of tens of thousands of SNPs found by GWAS to be at least weakly statistically related to the trait of interest [2, 3]. Models are generally fit using variants of linear or logistic regression. While current GPS are able to derive weak correlations between phenotype and genotype for traits like height, explaining about 10% of the observed variance [1], progress in predicting psychiatric disorders has been slower. For example, Levine et al. attempted to build a predictive score for depression using 11 SNPs identified as highly significant by GWAS using multinomial logistic regression, and ultimately explained less than 0.25% of the variance in observed depressive symptoms [4]. Taking a contrasting approach, Krapohl et al. [2] built GPS models for a number of traits, including some psychiatric conditions, using tens of thousands of SNPs chosen using a low GWAS significance threshold. However, these researchers were not able to find significant genotype-phenotype correlations for any of their psychiatric GPS models.

3 Dataset and pre-processing

The Philadelphia Neurodevelopmental Cohort is a large community sample collected from 2010 to 2012, containing medical information collected from a set of almost 10,000 patients between the ages of 8 and 21 who received care at the Children's Hospital of Philadelphia [5, 6]. We focused on a subset of 8179 patients who were both genotyped on Illumina SNP arrays and psychiatrically evaluated using GOASSESS, a computerized tool for assessing a wide range of psychiatric traits [5]. Our dataset contains 265267 measured SNPs for each subject; following the standard PLINK format `http://pngu.mgh.harvard.edu/~purcell/plink/data.shtml`, each SNP is encoded with two values, one for each allele. The GOASSESS results for each subject are encoded as a vector of numeric values, representing e.g. responses to questions or response times on cognitive tasks. Dropping demographic and age information from this data set left 735 values for each subject from the GOASSESS dataset.

3.1 SNP prioritization with functional genomics information

Because the number of SNPs measured is very large relative to the number of subjects, training a model on the full dataset would likely be intractable due to overfitting. We attempted to select a subset of SNPs likely to have an impact on phenotype using functional genomics data. "Open chromatin" regions of the genome – regions which are accessible to DNA-binding proteins (rather than being tightly wrapped by nucleosomes) – are enriched for functional elements [7]. We downloaded a dataset generated as part of the ENCODE Consortium [8] which describes genomic regions found to be open chromatin by a high-throughput DNAse I digestion hypersensitivity assay conducted on 97 different cell lines. Using an R script, we tested whether or not each SNP-cell line pair was accessible. For each SNP, we determined the number of cell lines for which it was reported as accessible (results shown in Fig. 1 for Chromosome 1).

We found that the majority of SNPs are in inaccessible regions; 76.3% were never accessible, while 83.2% were accessible in at most one cell type. Because the dataset's creators chose a false discovery rate of 1% when calculating open chromatin regions, SNPs found to be accessible in only one cell type may be the result of false positives. Selecting only SNPs which were accessible in 2 or more cell types left us with 44,360 SNPs.

To reduce the SNP set further, we downloaded annotation tracks for the hg19 reference human genome from the UCSC genome browser [9] defining exons (segments of DNA which have been observed in mature mRNAs) and defining the protein-coding regions of the genomes – the bases which directly encode the amino acids that make up proteins. (The hg19 genome was chosen as it had been used to assign genomic locations to the genotyped SNPs.) This last set contained only 5406 SNPs; unlike all others tested, it contained fewer SNPs than the number of subjects in our training set (Fig. 2). The protein-coding SNP set was used in all subsequent models.

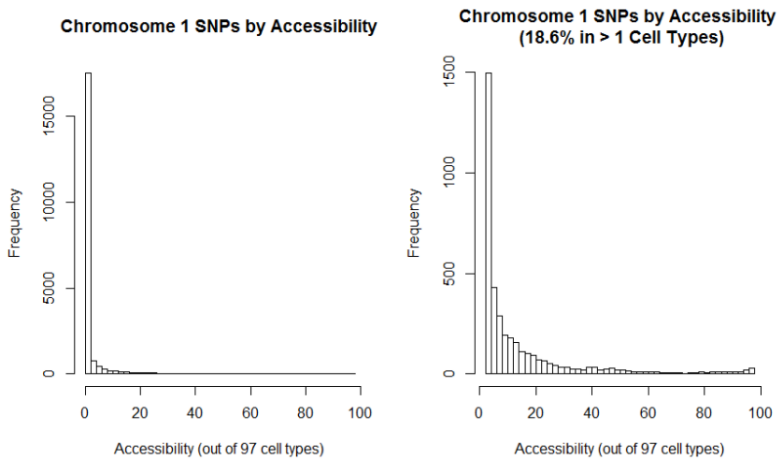

Figure 1: Chromosome 1 SNPs by Accessibility. Left panel – all SNPs; right panel – SNPs accessible in more than one cell type.

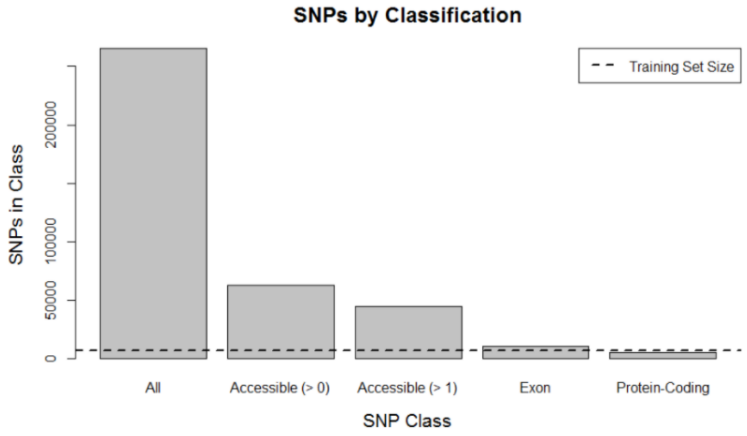

Figure 2: Sizes of identified SNP subsets. The bars 'Accessible (>0)' and 'Accessible (>1)' indicate the number of SNPs accessible in more than 0 and more than 1 ENCODE cell types, respectively.

3.2 Final preparation for model training

SNP data was one-hot encoded (with one value for each SNP allele, following the PLINK format). Some models were also trained to predict depression status using the first item in the depression subsection of GOASSESS as a proxy, or to predict the "MED score", a score ranking the patient's overall medical status which was included in the original patient data along with the GOASSESS data.

We divided these subjects into training, validation, and testing subsets with 90% training, 5% validation and 5% test, and used a consistent division of subjects in all models.

3.3 Label Dimensionality Reduction Using an Autoencoder

The filtered down GOASSESS dataset contained answers to 735 questions for each participant in the study. Many questions in the quiz are nested in structure, with one high level question and multiple follow up questions. There were also multiple questions regarding the same / similar phenotypes. The resulting dataset was highly correlated and compressible. In order to make this data useful as the

prediction output for classifiers, we decided to use a lower dimensional representation of the data. To do this, we used an autoencoder to learn a compressed representation of the data, which would then be used as the output of our classifiers. We can use the corresponding trained decoder and the predicted values of the compressed dataset to retrieve the predicted responses to the GOASSESS questions.

An autoencoder neural network is an unsupervised learning algorithm that applies backpropagation, setting the target values to be equal to the inputs. It tries to learn an approximation to the identity function.

Additionally, by placing constraints on the number of hidden network, such as by limiting the number of hidden units, we can discover the underlying structure about the data. But if there is structure in the data, for example, if some of the input features are correlated, then the autoencoder is able to discover them, and ends up learning a low-dimensional representation very similar to PCAs [10].

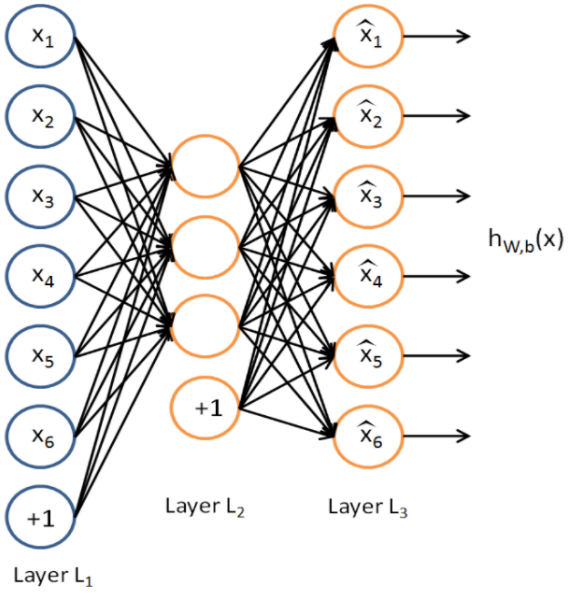

Figure 3: Autoencoder architecture.

A 5-layer fully-connected autoencoder was used with the 735 values per subject as the input. Binary-crossentropy was used as the loss metric, and the network was optimized using Adagrad, with a batch size of 256, and a train/validation split of 85/15. The encoding dimension was set to 30 after studying the relationship on the overall loss with increasing the dimension from 5 to 100. 30 was picked as the point where the loss plateaued. We used early stopping to avoid overfitting. The fully trained autoencoder yielded a validation loss of 0.12, which we compared to a baseline PCA implementation with the same encoded dimension size. The baseline implementation achieved a validation loss of 0.23.

4 Methodology

4.1 Recurrent Neural Network (RNN)

4.1.1 General Approach

We implemented four different RNN models to predict the MED classes and the autoencoder labels. The input to each RNN is the genomic data, the output of the last RNN unit is used to make prediction. To generate the input data, we first concatenate the reduced SNP data of each chromosome into one input data sequence, we then split this data sequence into many equally-sized segments (54 segments of 204 SNPs); each piece is feed into the RNN sequentially. Drawing an analogy from natural

language processing models, we are treating the entire dataset as a paragraph composed of words, and each piece as word embedding. We use root mean square error (RMSE) as the loss function for the autoencoder label prediction and cross entropy loss for MED class prediction.

4.1.2 Model

The RNN is an extremely expressive model that learns highly complex relationships from a sequence of data. The RNN maintains a vector of activation units for each element in the data sequence, this makes RNN very deep; the depth of RNN lead to two well-known issues, the exploding and the vanishing gradient problem [11, 12]. The exploding gradient problem is commonly solved by enforcing a hard constraint over the norm of the gradient [13]; the vanishing gradient problem is typically addressed by Gated Recurrent Unit (GRU) or Long Short-Term Memory (LSTM) activation architectures [14, 15, 16]. Both the LSTM and the GRU solves the vanishing gradient problem by re-parameterizing the RNN; The input to the LSTM cell is multiplied by the activation of an input gate, and the previous values are multiplied by a forget gate, the network only interacts with the LSTM unit via gates. GRU simplifies the LSTM architecture by combining the forget and input gates into an update gate and merging the cell state with the hidden state. We begin by evaluate the performance of GRU and LSTM activation unit on a unidirectional RNN, we then enhance our model's ability to capture contextual information by adding bi-directionality. Bidirectional RNN consists of a forward and a backward RNN structure. In the forward RNN, the input sequence is arranged from the first input to the last, the model computes a sequence of forward hidden states. The backward RNN takes the input sequence in reverse order, resulting in a sequence of backward hidden states. We concatenate output from both RNNs to make the final prediction. (Fig. 4)

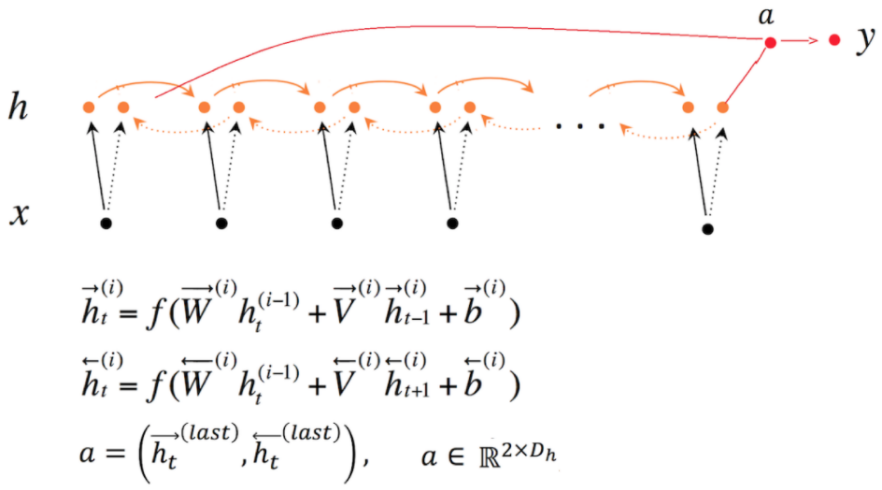

Figure 4: Bidirectional RNN.

The bidirectional RNN model must propagate dependencies over long distance to make the final prediction. The last layer of the network must capture all information from previous states, this is a challenging task for long sequential input. To overcome this bottleneck of information flow we implemented an attention mechanism inspired by recent results in natural language and image processing tasks. [17, 18, 19, 20] The attention-based model is an extension of the bidirectional RNN structure, the hidden state of each forward and backward hidden layer is concatenated into a single output vector, this concatenated vector is transformed into a scalar value via a set of attention weight vectors. The resulting scalar value from each hidden state is concatenated into a new vector, this vector goes through an additional projection layer to generate the final prediction. (Fig. 5).

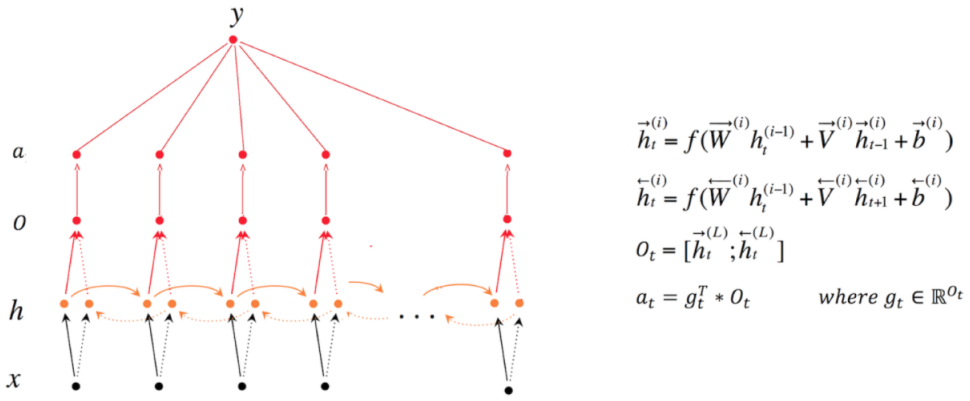

Figure 5: Bidirectional RNN with Attention Mechanism.

4.2 Feed Forward Neural Networks (FF)

4.2.1 General Approach

We implemented a 6-layer feed forward network to predict the MED classes and the autoencoder labels as well. The reason for using FF nets was to test the hypothesis of whether labels are predicted by a set of local SNPs or not. For the former we used RNNs and for the latter, we used FF nets. The inputs to the net is the filtered SNP data and the outputs are either autoencoder labels or MED data. The input data had about 10k features (5k major and 5k minor alleles). For consistency, we use root mean square error (RMSE) as the loss function for the autoencoder label prediction and cross entropy loss for MED class prediction.

4.2.2 Model

The input layer of our network had about 10k features. The size of the hidden layers reduced by a half at each step, with the final layer having 30 or 5 units, depending on the output data. The cost function was either MSE for the regression problem (autoencoder labels) or cross entropy (MED classes). The nonlinearity used was ReLU. We used the Xavier method to initialize our weights (Gaussian weights scaled by 1/fan-in). To avoid overfitting of our model, we used heavy regularization of our weights (by a factor of 10^3) and used dropouts of 50%. For MED class prediction, we had a training imbalance problem, which we attempted to address in two different ways. First was to modify our minibatch sampling, such that the samples in a minibatch have a uniform distribution of labels. The second method was reweigh the cross entropy cost function to be more sensitive to the labels that are less represented in the data. These methods force the model to have better prediction across all classes, rather than focusing on just the dominant class.

5 Results

5.1 RNN

In the unidirectional RNN case, the LSTM structure performed slightly better than GRU on both tasks, we used LSTM as the activation unit in the bidirectional and the attention model. The Bidirectional RNN is slightly more accurate than the unidirectional RNN on the MED classification task. The attention model with bidirectional LSTM yields the best MED prediction, however the attention model did not help in predicting the autoencoder label. The prediction results seem to indicate that the sequential information embedded in the SNPs helped the prediction of MED classes. The results are listed in Table 1.

Table 1: Results of using RNNs

Model	MED accuracy	Autoencoder label loss
Baseline (regression)	0.25	0.010
Unidirectional GRU	0.272	0.089
Unidirectional LSTM	0.281	0.085
Bidirectional LSTM	0.288	0.087
Bidirectional LSTM with Attention	0.303	0.090

5.2 Feed Forward Neural Networks

The feedforward network performed very similarly to the RNN. The model massively overfit without regularization and dropouts. In our experiments, reweighing the loss function with weights inversely proportional to the representation size of the class outperformed balanced sampling during minibatches when it came to measuring mean accuracy across classes. If we chose to measure just accuracy (and not the mean accuracy across labels), then using the imbalanced data without changing batch sampling or weights of the loss function produced the best results, which were more or less in line with the linear regression accuracies. The results are shown in Table 2. It should be noted that variations of balancing the dataset were only useful for the MED prediction as those labels were unbalanced. The autoencoder labels were continuous values in high dimensions, so it was difficult to determine the balance of that network.

Table 2: Results of using feedforward networks. Vanilla model indicates the feedforward network with regularization

Model	MED mean accuracy	Autoencoder label loss
Vanilla	0.22	0.009
Vanilla + balanced sampling	0.24	0.009
Vanilla + weighted loss	0.27	0.009

6 Conclusion and Future Work

We used an autoencoder to significantly reduce the dimensionality of our labels and used RNNs and FFs to predict those labels using a filtered set of SNPs. We also used our neural nets to predict the medical condition of the patients. While our autoencoder successfully reduced the size of the labels with minimal loss, our neural nets had at best slight gains over using linear regression. We attribute this to multiple reasons. Firstly, genotypic data is not an exact predictor of phenotypic data, and secondly, with the limited number of samples we have, we were forced to reduce the dimensionality of our features (SNPs) and labels very aggressively. This reduction comes at a cost of removing possibly predictive and useful information from our data.

While our models cannot reliably distinguish healthy from affected subjects, the fact that they outperform baseline models suggests that they identified some SNPs associated with the conditions we investigated. In the future, we could use ablation experiments or attention mechanisms to identify the relevant SNPs. Additionally, unlike conventional linear models, our approach allows for relationships between SNPs to be identified. We could also improve the models by finding a more sophisticated way to incorporate the chromatin accessibility data. For example, SNPs accessible in neural cells may be more relevant to psychiatric phenotypes. A properly designed model could potentially learn how chromatin accessibility affects SNP relevance directly. Other types of functional genomics data, such as genome-wide three-dimensional chromatin structure as captured by the 5C technique [21], could also be incorporated to improve the model. In the future, we could also use fully imputed SNP data to increase our pool of potentially relevant SNPS.

In recent years, the price of DNA sequencing has dropped at a rate outperforming Moore's Law [22]. If this trend continues, future genomic datasets could contain many more subjects than is currently feasible. Since conventional GWAS studies already gain power as the number of subjects increases, there is every reason to believe our approach could similarly improve on a larger dataset, while retaining the advantages neural networks have over conventional linear models.

References

[1] Lango allen H, Estrada K, Lettre G, et al. Hundreds of variants clustered in genomic loci and biological pathways affect human height. *Nature*. 2010;467(7317):832-8.

[2] Krapohl E, Euesden J, Zabaneh D, et al. Phenome-wide analysis of genome-wide polygenic scores. *Mol Psychiatry*. 2016;21(9):1188-93.

[3] Dudbridge F. Power and predictive accuracy of polygenic risk scores. *PLoS Genet*. 2013;9(3):e1003348.

[4] Levine ME, Crimmins EM, Prescott CA, Phillips D, Arpawong TE, Lee J. A polygenic risk score associated with measures of depressive symptoms among older adults. *Biodemography Soc Biol. 2014*;60(2):199-211.

[5] Gur RC, Richard J, Calkins ME, et al. Age group and sex differences in performance on a computerized neurocognitive battery in children age 8-21. *Neuropsychology*. 2012;26(2):251-65.

[6] Satterthwaite TD, Connolly JJ, Ruparel K, et al. The Philadelphia Neurodevelopmental Cohort: A publicly available resource for the study of normal and abnormal brain development in youth. *Neuroimage*. 2016;124(Pt B):1115-9.

[7] Tsompana M, Buck MJ. Chromatin accessibility: a window into the genome. *Epigenetics Chromatin*. 2014;7(1):33.

[8] Thurman RE, Rynes E, Humbert R, et al. The accessible chromatin landscape of the human genome. *Nature*. 2012;489(7414):75-82.

[9] Rosenbloom KR, Armstrong J, Barber GP, et al. The UCSC Genome Browser database: 2015 update. Nucleic Acids Res. 2015;43(Database issue):D670-81.

[10] "Unsupervised Feature Learning and Deep Learning Tutorial." *Unsupervised Feature Learning and Deep Learning Tutorial*. N.p., n.d. Web. 15 Dec. 2016.

[11] Bengio, Yoshua, Simard, Patrice, Frasconi, Paolo, 1994. Learning long-term dependencies with gradient descent is difficult. *Neural Networks, IEEE Transactions on*, 5, pp.157?166.

[12] Jozefowicz, Rafal, Zaremba, Wojciech, and Sutskever, Ilya. An empirical exploration of recurrent network architectures. *In Proceedings of the 32nd International Conference on Machine Learning (ICML-15)*, pp. 2342? 2350, 2015.

[13] Pascanu, Razvan, Mikolov, Tomas, and Bengio, Yoshua. On the difficulty of training recurrent neural networks. *arXiv preprint* arXiv:1211.5063, 2012.

[14] Hochreiter, S., & Schmidhuber, J. (1997). Long short-term memory. *Neural Computation*, 9(8), 1735?1780.

[15] Gers, F., Schraudolph, N., & Schmidhuber, J. (2002). Learning precise timing with LSTM recurrent networks. *Journal of Machine Learning Research*, 3, 115?143.

[16] Cho, Kyunghyun, van Merrienboer, Bart, Gulcehre, Caglar, Bougares, Fethi, Schwenk, Holger, and Bengio, Yoshua. Learning phrase representations using rnn encoder- decoder for statistical

[17] Bahdanau, Dzmitry, Kyunghyun Cho, and Yoshua Bengio. "Neural machine translation by jointly learning to align and translate." *arXiv preprint* arXiv:1409.0473 (2014).

[18] Mnih, Volodymyr, Nicolas Heess, and Alex Graves. "Recurrent models of visual attention." *Advances in Neural Information Processing Systems*. 2014.

[19] Karol Gregor, Ivo Danihelka, Alex Graves, and Daan Wierstra. DRAW: A recurrent neural network for image generation. *CoRR*, abs/1502.04623, 2015.

[20] Hermann, Karl Moritz, et al. "Teaching machines to read and comprehend." *Advances in Neural Information Processing Systems*. 2015.

[21] Dostie J, Richmond TA, Arnaout RA, et al. Chromosome Conformation Capture Carbon Copy (5C): a massively parallel solution for mapping interactions between genomic elements. *Genome Res. 2006*;16(10):1299-309.

[22] `https://www.genome.gov/27565109/the-cost-of-sequencing-a-human-genome/`