Using Deep Learning to Predict Customer Churn in a Mobile Telecommunication Network

Federico Castanedo
wiseathena.com
CA 94105 USA

Gabriel Valverde
wiseathena.com
CA 94105 USA

Jaime Zaratiegui
wiseathena.com
CA 94105 USA

Alfonso Vazquez
wiseathena.com
CA 94105 USA

Abstract

Customer churn is defined as the loss of customers because they move out to competitors. It is an expensive problem in many industries since acquiring new customers costs five to six times more than retaining existing ones [1-4]. In particular, in telecommunication companies, churn costs roughly $10 billion per year [5]. A wide range of supervised machine learning classifiers have been developed to predict customer churn [6-9]. In general, these models base their effectiveness in the feature engineering process which is usually time consuming and thus tailored to specific datasets. Since deep learning automatically comes up with good features and representation for the input data; we investigated the application of autoencoders, deep belief networks and multi-layer feedforward networks with different configurations. We report results for predicting customer churn using a four-layer feedforward architecture. To scale the model to full-sized high dimensional customer data, like the social graph of a customer, we introduced a data representation architecture that allows efficient learning across multiple layers of detailed user behavior representations. In this article, we use billions of call records from an enterprise business intelligence system and present our current work towards using deep learning for predicting churn in a prepaid mobile telecommunication network. To the best of our knowledge this is the first work reporting the use of deep learning for predicting customer churn. On average, our model achieves 77.9% AUC on validation data, significantly better than our prior best performance of 73.2% obtained with random forests and an extensive custom feature engineering applied to the same datasets.

1 Introduction

Customer churn is a fundamental problem for companies and it is defined as the loss of customers because they move out to competitors. Being able to predict customer churn in advance, provides to a company a high valuable insight in order to retain and increase their customer base. A wide range of customer churn predictive models has been developed in the last years. Most advanced models make use of state-of-the-art machine learning classifiers such as random forests [6][10]. Machine learning classifiers work well if there is enough human effort spent in feature engineering, so it is possible to find a reasonable boundary of the classes in feature space. Thus having the right features for each particular problem is usually the most important thing.

To solve these problems companies spent a lot of feature engineering effort designing specific features for problems like churn prediction and fraud detection. Even more problematic, features obtained in this human feature engineering process are usually over-specified and incomplete. Feature engineering becomes not optimal in companies that have huge amounts of data. For instance, in some telecommunication companies the data warehouse system holds more than 100000 variables.

Since deep learning attempts to learn multiple levels of representation and automatically comes up with good features and representation for the input data, we have recently investigated the application of deep learning in predicting customer churn in prepaid mobile telecommunication networks.

Our main motivation to investigate and consider the application of deep learning as a predictive model is to avoid time-consuming feature engineering effort and ideally to increase the predictive performance of previous models (or at least not degrade).

Well known companies are also reporting the use of deep learning in commercial products. Microsoft work on Mavis speech recognition system represents one of the first examples of using a deep neural network in a commercial product [11]. IBM and Google are also using deep learning models for speech recognition [12][13] and image processing. Deep learning has been also applied to different domains such as automatic music recommendation [14] and prediction of protein structure [15]. However, to the best of our knowledge this is the first work reporting the use of deep learning for predicting churn in a mobile telecommunication network.

To investigate the feasibility of using deep learning models in production we trained and validated the models using large-scale historical data from a telecommunication company with $\approx$1.2 million customers and span over sixteen months. This dataset is extremely challenging because churn rate is very high and all customers are prepaid users, so there is no specific date about contract termination and this action must be inferred in advance from similar behaviors. Figure 1 depicts a fragment of the input data as a graph representation, it can be seen that there are complex underlying interactions amongst the users.

In this workshop paper, we present preliminary results from analyzing billions of call records for the problem of predicting churn using a deep neural network. Obtained results are very promising and open a novel research area to consider the use of deep learning models in production for predicting customer churn or other similar problems, like fraud detection.

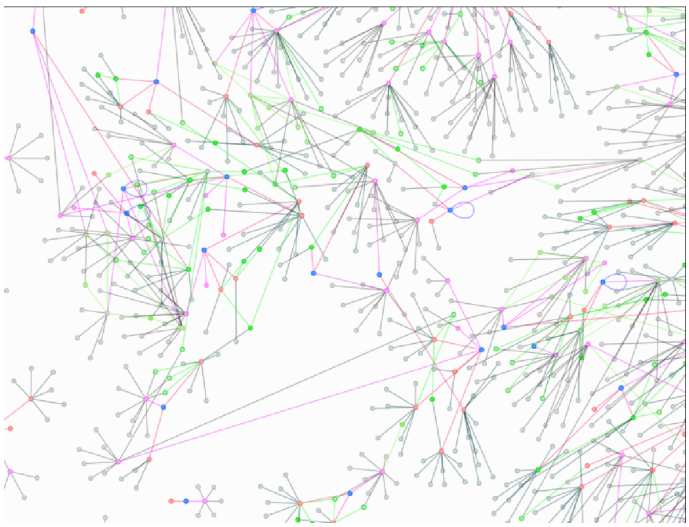

Figure 1: A real graph representation of telecommunication data where each node identifies a device (IMEI) or a phone number (SIM) and edges represent different type of interactions among them.

2 Churn prediction in prepaid mobile telecommunication network

Mobile telecommunications markets across the world are approaching saturation levels. Therefore the current focus is to move from customer acquisition towards customer retention. Having a good churn prediction model becomes extremely useful in order to minimize the churn rate because tailored promotions can be offered to specific customers that are not satisfied [7]. Churn in prepaid services is actually measured based on the lack of activity in the network over a period of time. Thus, there is no formal notification from the customer of ending a contract term. Our goal is to

infer when this lack of activity may happen in the future for each active customer. The sooner these changing patterns are detected the more opportunities and time the company will have to retain the customer.

Previous works about predicting churn in telecommunication networks use graph processing techniques [8]. They identified already churned customers and analyzed a graph model to infer interactions with the current customers with the aim of predicting new churners based on these interactions. Others predict customer churn by analyzing the interactions between the customer and the Customer Relationship Management (CRM) data [9].

In our approach, churn prediction can be viewed as a supervised classification problem where the behavior of previously known churners and non-churners are used to train a binary classifier. During the prediction phase new users are introduced in the model and the likelihood of becoming a churner is obtained. Depending on the balance replenishment events, each customer can be in one of the following states: (i) new, (ii) active, (iii) inactive or (iv) churn (see Figure 2). Customer churn is always preceded by an inactive state and since our goal is to predict churn we will use future inactive state as a proxy to predict churn. In particular, we define t=30 days without doing a balance replenishment event as the threshold used to change the state from active to inactive.

Depending on the balance replenishment events, each customer can be in one of the following states: (i) new, (ii) active, (iii) inactive or (iv) churn (see Figure 2). Customer churn is always preceded by an inactive state and since our goal is to predict churn we will use future inactive state as a proxy to predict churn. In particular, we define t=30 days without doing a balance replenishment event as the threshold used to change the state from active to inactive [1].

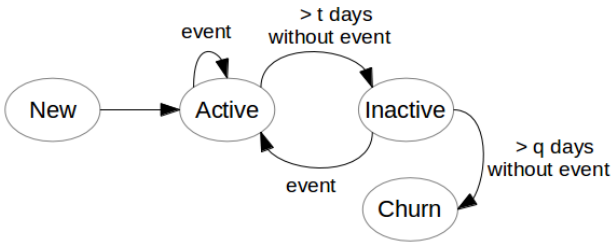

Figure 2: Diagram of possible customer states based on the balance replenishment events and specific thresholds t and q.

3 Deep learning models for churn prediction

Our sensory inputs for the deep learning model are based on the actions or events carried out by the phone users. More in detail we have two main sources of information: Call Detail Records (CDR) and balance replenishment records.

3.1 Input data and preparation

Each CDR provides a detailed record about each call made by the customer, having (at least) the following information:

- Id_Cell_Start: Id of the cell tower where the call is originated.
- Number_A: Phone number of the customer originating the call.
- Id_Cell_End: Id of the cell tower where the call is finished.
- Number_B: Destination number of the call.
- Timestamp: Time-stamp of the beginning of the call.

[1] This is a common metric used in telecommunication, even if the deactivation of the phone number happens several days days after (i.e. 180 days).

- Duration: Call duration (secs).
- IMEI: Unique identification number of the phone terminal.
- Type: Incoming or outgoing call.

Due to confidentiality reasons and to respect privacy both Number_A and Number_B are encrypted. On the other hand, balance replenishment records has (at least) the following information:

- Number: Phone number related to the balance replenishment event.
- Timestamp: Time-stamp of the balance replenishment event.
- Amount: Amount of money the customer spent in the balance replenishment event.

We compute one input vector per user-id for each month. This input vector contains both calls events and balance replenishment history of each customer. If we consider an user-user adjacency matrix, it will be extremely huge (roughly 1.2M x 1.2M entries) and does not provide valuable signals for most users. To relax the input representation problem we consider only call records for each top-3 users[2] (although some kind of low-rank approximation can be applied as well [16][17]). These top-3 users may be binding to to the same company or not. We followed an approach similar to Bag-of-Words and generated a 48-dimensional vector X where each position refers to the sum of total seconds each user spent on the phone in that 30 minutes time interval over the complete month, so we will have a 145-dimensional input vector. In Figure 6 we shown an example of the mobile-call fingerprint computed.

Then, we add another 48-dimensional vector per user with the total amount of cash spent in each monthly slot. Next we include 5 features specific to the business and due to the confidentiality of the data we can not provide details of them. But we can report they do not require any extensive feature engineering process. Finally we add the binary class indicating if the user will be active in month $M + 1$ or not and end up with a 199-dimensional input vector X.

3.2 Deep learning model

According to Cybenko theorem, being ϕ any sigmoid function (such as $\tanh$), given any compact set in R^n and $\varepsilon > 0$ there are some vectors $w_1, w_2, ...w_N, \alpha, b$ and a parametric function $G(\cdot, w, \alpha, b)$ from the compact set to R given that:

$$\mid G(x, w, \alpha, b) - f(x) \mid < \varepsilon, \forall x \in K \tag{1}$$

where K is compact set in R^n and

$$G(x, w, \alpha, b) = \sum_{i=1}^{N} \alpha_i \phi_i(w_i^T x + b_i) \tag{2}$$

and $w_i \in R^n, \alpha_i, b_i \in R, w = (w_1, w_2, ..., w_N), \alpha = (\alpha_1, \alpha_2, ...\alpha_N)$ and $b = (b_1, b_2, ..., b_N)$.

This theorem states that a neural network with a single hidden layer is able to approximate any function with a specific precision ε. It can be extended to several hidden layers which add more levels of abstraction and increase model capabilities [19]. We use a standard multi-layer feedforward architecture were the weighted combination $\alpha = \sum_{j=1}^{n} w_j x_j + b$ of the n input signals is aggregated, and then an output signal $f(\alpha)$ is transmitted by the connected neuron. The function f used for the nonlinear activation is the $\tanh$. Each layer ℓ computes an output vector z_ℓ using the output $z_{\ell-1}$ of the previous layer, starting with the input z_0 by applying:

$$z_\ell = \tanh(W_\ell Z_{\ell-1} + b_\ell) \tag{3}$$

were b_ℓ refer to the column vector of biases for layer ℓ and W_ℓ denotes the weight matrix connecting layers $\ell - 1$ and ℓ. The last output layer Z_L generates the predictions by applying the softmax function:

$$z_{Li} = \frac{e^{b_{Li} + W_{Li} z_{L-1}}}{\sum_{j=1}^{n} e^{b_{Lj} + W_{Lj} z_{L-1}}} \tag{4}$$

[2]This is just a particular case of top-K users and different K-values may be used.

where W_{Li} is the i-th row of W_L, z_{Li} is positive and $\sum_i z_{Li} = 1$. Since the output z_{Li} is an estimator of $P(Y = i|j)$ being $Y = i$ the i-th class associated with input pattern j, it can be used to minimize the loss function $L(z_L, y)$ for each training example j. In particular, we use the negative conditional log-likelihood $L(z_L, y) = -\log z_{Ly}$ as a loss function whose expected value over pairs (j, y) is minimized. To minimize the loss function we apply a standard stochastic gradient descendent (SGD) with the gradient computed via backpropagation. For the initial weight distribution we employ a random initialization drawn from a normal distribution with 0.8 standard deviation.

We use dropout as a regularization technique to prevent over-fitting [12][18]. For the input layer we use the value of 0.1 and 0.2 for the hidden layers. By using dropout each training example trains a different model but they share the same global parameters and allows models to be averaged as ensembles, preventing over-fitting while improving generalization.

4 Experiments

Our deep neural network is trained to distinguish between active and inactive customers based on learned features associated with them. In the training phase each instance contains input data for each customer together with the known state in the following month (see Figure 3). In the validation phase data from the next month is introduced into the model and the prediction errors are computed.

Using training data of month M_i (tagged with the state at the end of month M_{i+1}) we learned a multi-layer feedforward network p_i. With the model p_i we can generate predictions for the end of month M_{i+2} using data from month M_{i+1} (see Figure 3).

It is important to clarify that each instance in the training and validation data may refer to different customers, so we are not taking the customer identification into consideration to generate the predictions.

In Table 1 we report the number of input records per month and the number of users going from active to inactive each month. The model has been evaluated over 12 months of real customer data (from March 2013 to February 2014), we trained 12 models and generate predictions with each model for all the months. In Figure 5 we show obtained Area Under the Curve results for all the trained models. It seems that we have some problems with the input data for the month of July 2013 since their values are extremely low for most of the trained models. At the time of this submission we are in the process of analyze further this issue. Nevertheless most of the generated models are quite stable along the different months.

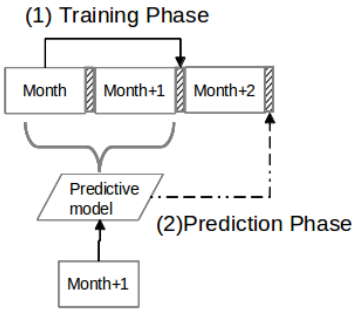

Figure 3: Training and validation phases. Input data from month m and each customer state in month $m+1$ is used to train a model p. In prediction phase, model p is used with month $m+1$ data to generate predictions for month $m+2$.

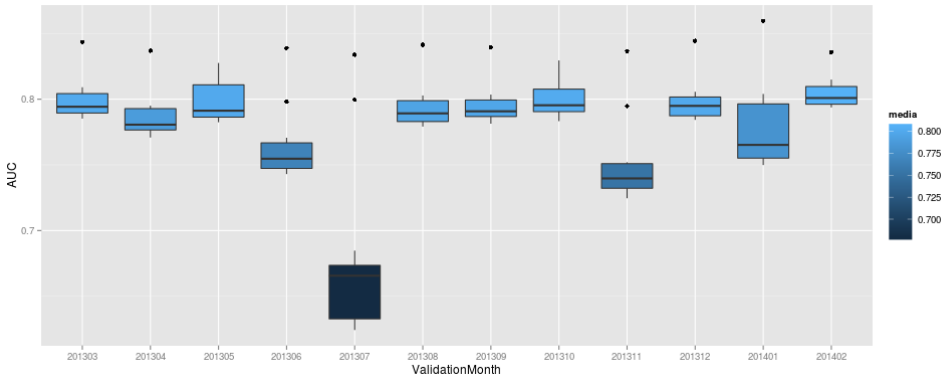

Figure 4: Boxplot of AUC values for each validation month using all trained models.

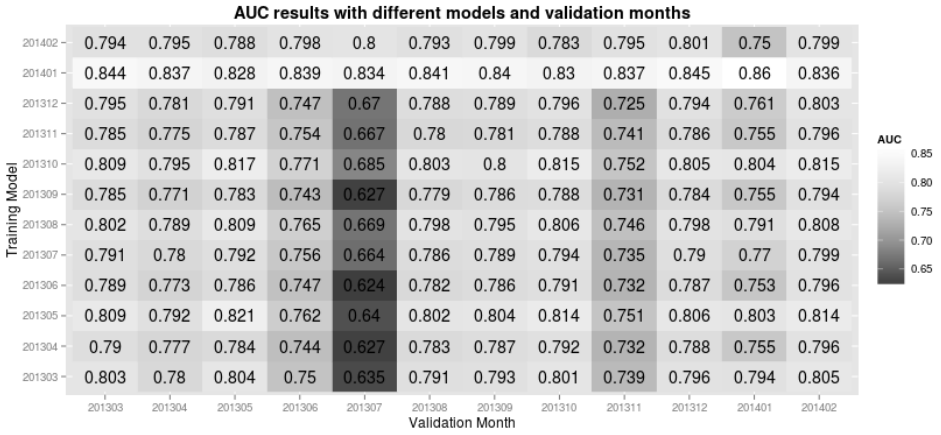

Figure 5: AUC results for each pair of trained model and validation month. Most of the generated models are quite stable along the different validation months.

Table 1: Input data figures. Total number of call records, balance replenishment events and ratio of active/inactive users in the next month.

Month	Call Records	Balance Events	% Active(M+1)	% Inactive(M+1)
March 2013	172.804.039	8.499.382	948.828 (79 %)	241.413 (21 %)
April 2013	164.847.311	8.352.734	924.833 (78 %)	250.019 (22 %)
May 2013	169.240.688	8.399.027	940.072 (79 %)	241.677 (21 %)
June 2013	166.815.828	8.004.702	931.172 (79 %)	246.248 (21 %)
July 2013	175.256.082	8.618.246	953.092 (77 %)	271.885 (23 %)
August 2013	175.780.713	8.470.981	948.131 (78 %)	258.447 (22 %)
September 2013	166.966.639	8.099.583	924.996 (78 %)	259.155 (22 %)
October 2013	171.762.970	8.312.791	935.338 (79 %)	245.855 (21 %)
November 2013	164.293.483	7.961.236	940.098 (80 %)	229.142 (20 %)
December 2013	187.114.263	9.120.727	975.610 (77 %)	287.387 (23 %)
January 2014	229.965.950	8.418.411	960.606 (79 %)	244.128 (21 %)
February 2014	207.292.944	7.528.277	904.561 (79 %)	240.348 (21 %)
March 2014	223.920.187	8.216.505	921.037 (78 %)	246.197 (22 %)
April 2014	213.184,319	7.848.203	894.899 (77 %)	263.470 (23 %)
May 2014	221.207.785	8.034.993	891.693 (77 %)	260.294 (23 %)
June 2014	207.199.280	7.643.465	877.358 (77 %)	251.341 (23 %)
July 2014	219.223.332	8.295.363	880.149 (76 %)	275.826 (24 %)

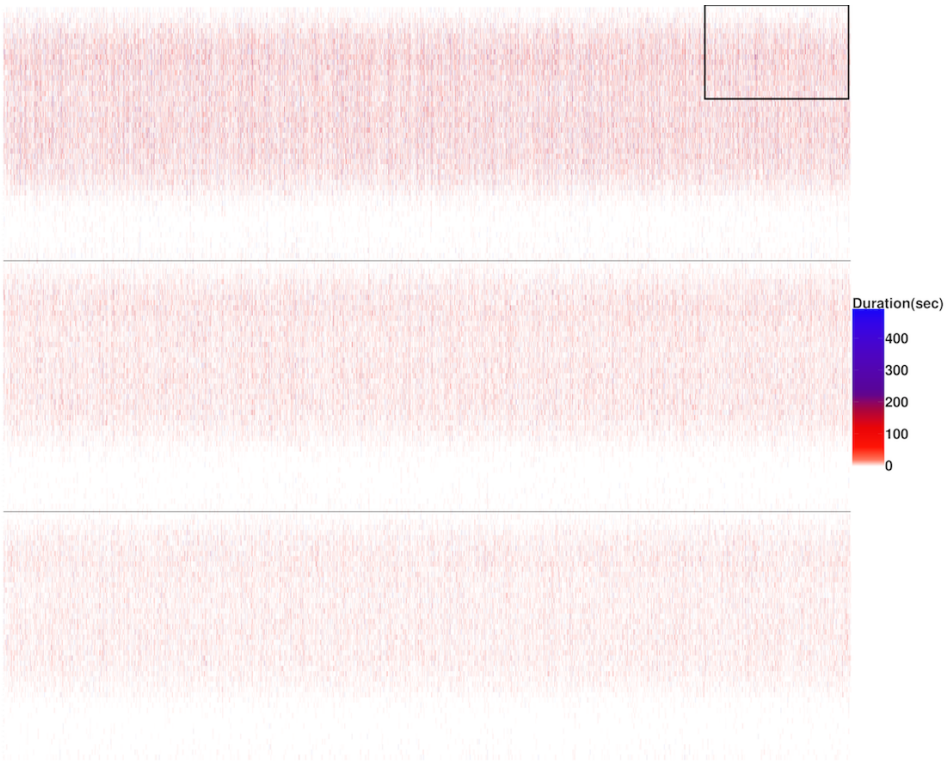

Figure 6: A subset of the high dimensional customer data. Pattern calls for each customer and per each time-slot for the top-3 most frequent links. It is clear that interactions with the top-1 link (top row) are higher than those with top-3 (bottom row). This representation allows deep learning models to be used with most telecommunication datasets (see section 3.1 for details). Marked area in black is shown in Figure 7.

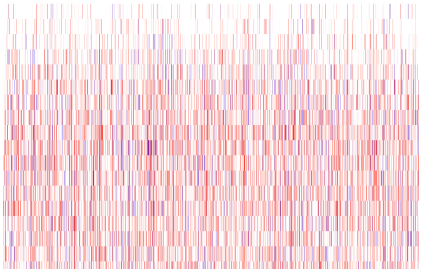

Figure 7: Detailed call pattern information corresponding to the marked area in Figure 6 used as input for our deep learning model.

5 Summary and future work

To apply deep learning models to predict customer churn prediction in a mobile telecommunication network, we implemented a multi-layer feedforward architecture and introduced a novel way to encode input data to learn hierarchical representation on real datasets.

Our experiments suggest multi-layer feedforward models are an effective algorithm for predicting churn and capture the complex dependency in the data. Experiments shown that the model is quite stable along different months, thus generalize well with future instances and do not overfit the training data.

While we demonstrated its success on churn prediction, such architecture can be potentially applied to other business intelligence prediction tasks like fraud detection or upsell.

For further development we are planning to include location data (latitude and longitude) of each call into the input data and hope to improve obtained results. Another open area of research is to study the application of deep belief networks in this scenario. Since, it has been shown that unsupervised pre-training improves the predictive performance [20].

One limitation of the current architecture is that the full user-user input data is extremely sparse and if we want to consider long-term user interactions it becomes very big. So, to better model long-term interactions in a telecommunication dataset, an input data architecture may encode also these long-term interactions amongst users.

References

[1] C. B. Bhattacharya. (1998) When customers are members: customer retention in paid membership contexts. *Journal of the Academy of Marketing Science*, 26 (1) 31-44.

[2] E. Rasmusson. (1999) Complaints Can Build Relationships. *Sales and Marketing Management*, 151 (9) 89-90.

[3] M. Colgate, K. Stewart, R. Kinsella. (1996) Customer defection: a study of the student market in Ireland. *International Journal of Bank Marketing*, 14 (3) 23-29.

[4] A. D. Athanassopoulos. (2000) Customer Satisfaction Cues To Support Market Segmentation and Explain Switching Behavior. *Journal of Business Research*, 47 (3), 191-207.

[5] P. Kisioglu & Y. I. Topcu. (2011) Applying Bayesian Belief Network approach to customer churn analysis: A case study on the telecom industry of Turkey. *Expert Systems with Applications*, 38 (6), 7151-7157.

[6] S. Yoon, J. Koehler & A. Ghobarah. (2010) Prediction of Advertiser Churn for Google AdWords. *JSM Proceedings*, American Statistical Association.

[7] B. Huanh, M.T. Kechadi & B. Buckley. (2013) Customer Churn Prediction in Telecommunications. *Expert Systems with Applications*, 39, 1414-1425.

[8] D. Chakraborty et. al. (2012) Method for Predicting Churners in a Telecommunications Network. *US Patent* 8194830 B2.

[9] B. Eilam, Y. Lubowich & H. Lam. (2013) Method and Apparatus for Predicting Customer Churn.*US Patent* 8615419 B2.

[10] L. Breiman. (2001) Random Forests.*Machine Learning*, 45 (1), 5-32.

[11] L. Deng et. al. (2013) Recent Advances in Deep Learning for Speech Research at Microsoft. *ICASSP*, 8604-8608.

[12] G. Dahl et. al. (2013) Improving Deep Neural Networks for LVCSR using Rectified Linear Units and Dropout. *ICASSP*, 8609-8613.

[13] G. Hinton et. al. (2012) Deep Neural Networks for Acoustic Modeling in Speech Recognition: The Shared Views of Four Research Groups. *IEEE Signal Processing Magazine*, 29 (6) 82-97.

[14] A. van den Oord, S. Dieleman & B. Schrauwen. (2013) Deep Content-based Music Recommendation. *Advances in Neural Information Processing Systems (NIPS)*, 2643-2651.

[15] J. Zhou, O. G. Troyanskaya. (2013) Deep Supervised and Convolutional Generative Stochastic Network for Protein Secondary Structure Prediction.*Advances in Neural Information Processing Systems (NIPS)*.

[16] D. Achlioptas & F. Mcsherry. (2007) Fast Computation of low-rank Matrix Approximations.*Journal of the ACM*, 54 (2).

[17] N. Srebro & T. Jaakkola. (2003) Weighted low-rank approximations. *ICML*, (3) 720–727.

[18] G. Hinton et. al. (2012) Improving Neural Networks by Preventing Co-adaptation of Feature Detectors. *arXiv preprint arXiv:1207.0580*

[19] K. Hornik, M. Stinchcombe & H. (1989) White. Multilayer feedforward networks are universal approximators. *Neural Networks*, 2 (5), 359-366.

[20] D. Erham. (2010) Why Does Unsupervised Pre-training Help Deep Learning?. *Journal of Machine Learning Research*, 11, 625-660.