# Machine Learning Engineer Nanodegree

## Capstone Proposal

Boris Terentiev October 4th, 2018

## Proposal

### Background Information

Deep Learning Algorithms have achieved incredible performance in domains like image and text recognition. Fortunately companies and researchers make the trained models publicly available and techniques like transfer learning make it possible to reuse this state of the art implementations as a building block for classification tasks on different data sets and even small data sets.

On the other hand real life data sets contain often more features than only images or only text, i.e. the dataset contains additionally categorical, continuous or raw sensory data. It is sensible to believe that over time more and more feature domains (e.g. seismometer data, fitness tracker data, etc.) will be covered by specialized Machine Learning Algorithms. That is we get a set of specialized trainable algorithms that perform exceptionally well for a single feature and can be trained with low computational cost.

Then naturally the following problem arises: It it possible to combine this specialized implementations to achieve better performance than every single feature classifier on its own. (In fact I have to tackle the very same problem at work right now in a domain I unfortunately can't disclose here)

In the lecture we have seen AdaBoost as a possibility to combine multiple classifiers in a computationally efficient manner. The in depth description from the lecture states that AdaBoost is a special case of an additive model. The document also states that it is computationally expensive to apply the additive model to multiple basis classifiers, but if the training for every basis classifier is cheap then it may still be sensible to combine multiple classifiers this way.

In this project I will use the MovieLens dataset to create multiple classifiers for the movie genre that can be trained cheaply on their own and try to combine them by means of an additive model to achieve an overall better classifier.

### Problem Statement

Movies are typically classified into genres, i.e. drama, comedy, thriller, etc. As a single movie can fit into more than one genre, e.g. Dramedy or Action and Fantasy, the problem of estimating the movie genres is a **multi label classification problem**. Every movie is attributed to at least one genre from 18 possible genres (in the MovieLens data). As stated in the previous paragraph the goal is to create multiple, **cheaply trainable classifiers** that will be **combined by an additive model**.

As we now know how to combine the basis classifiers, the next natural question is: What are the basis classifiers?

**Basis classifier 1: Movie posters**

For marketing purposes movie studios create a plethora of marketing material such as trailers, posters and merchandise with the intention to show how awesome their newest production is, but also to give a sense what to expect from the movie, i.e. especially to transport a sense for the movie genre. Humans are trained to distinguish the little cues in a poster, e.g. smiling people, a dark tone of the poster or the appearance of Jack Black to understand the sentiment and ultimately the genre.

We have learned in class how to create your own image classification algorithms that work on even small datasets. The secret ingredients are deep neural nets (DNN) that were trained by researchers or corporations on the ImageNet dataset and can be reused for other datasets by means of transfer learning.

There is one difference to the application in udacity class though. The dog classification problem was a multi class classification problem, that is the image could only belong to a single class. The current problem is a multi label classification problem. This means we **can't use accuracy as performance indicator** of the model but have to use a different performance metric, e.g. **the hamming loss** (defined on wikipedia)

**Basis classifier 2 and 3: Tags and cast**

The input for the second classifier will be either **movie tags** or **movie cast**. Using the movie cast makes sense, as from experience some actors tend to be part of dramas and some actors tend to participate comedies. For example a linear regression could be able to catch this correlations and also generalize for rather exotic combinations of actors. On the other hand every individual actor doesn't participate in hundreds of movies but rather a hand full. Therefore it is hard to say if the number of examples per actor is large enough to learn the correlations.

If the approach with the actors doesn't work, I will try the same approach with movie tags.

## Datasets and Inputs

MovieLens is a non commercial, personalized movie recommendation system and relies on the users to provide metadata for movies. The project provides extensive datasets meant for research or education, such as tags generated by the users and links to imdb and tmdb for a huge set of movies. Additionally one can use the links to tmdb and imdb to retrieve additional information about specific movies, such as numerous posters per movie and missing metadata, e.g. the cast.

As stated in the previous section, I am going to use the movie posters as the input for the first basis classifier and the tags or cast as input for the second basis classifier.

The movie posters are images that can be processed by the pretrained models, e.g. VGG16, InceptionV3,..., when we just apply a normalization to the three channel, two dimensional integer matrices.

The tags and cast are lists of categorical values. One can map this lists to binary vectors where a value is 1 if the movie belongs to the respective category.

The MovieLens data is unbalanced as it is a real world collection of different movies and some genres are more common than others. A model trained on such data could lean towards more common classes without learning the abstract properties to distinguish the genres accurately.

## Solution Statement

The first goal is to examine if a deep neural network (DNN) trained by means of transfer learning is able to get a sense of the cues in a movie posters and to make better predictions than a random model. A possible problem may be that poster styles change over time significantly, e.g. a poster for Ben Hur from 1959 could be a poster for a light-hearted comedy of today. Therefore it may be sensible to restrict the time frame.

Then I will try to create classifier based on movie metadata, specifically on the user-generated tags or cast.

In a final step I am going to combine this classifiers into a single classifier, either by means additive model.

## Benchmark Model

Preliminary analysis of the dataset showed that the movies are assigned to 1.9 genres on average with 18 genres in total. Additionally our dataset is unbalanced, i.e. some genres occur more often than others. Therefore a sensible trivial **benchmark model is to assign two most frequent genres** to every movie.

## Evaluation Metrics

As stated before the problem of assigning movie genres is a **multi label classification problem**. One possible metric to evaluate the performance of a model that solves multi label classification problem is the **hamming loss**.

## Project Design

First I have to enrich the MovieLens dataset with the movie posters. I will do this by using the links to tmdb, provided in the MovieLens datasets.

Next step is to analyze and filter the data for movies that can't be processed, e.g. no poster, no genres, etc.

Next we are going to split the data into training, validation and testing sets.

Next I am going to implement the benchmark model and evaluate the performance metric for the benchmark model.

Then we will create a classifier based solely on posters. This classifier will be a deep neural network (DNN) and will be trained by means of transfer learning. The framework of choice will be Keras as it makes the application of transfer learning simple. The training loss will be cross entropy.

I will choose the best model by applying the hamming loss to the validation set and assess the performance of the final model by applying the hamming loss to the test set.

Next step will be to create classifiers that use movie metadata.

Again the performance of the classifiers is assessed by calculating the hamming loss of the test set.

Next I will choose a way (e.g. by adding additional neuron layers) to combine the resulting classifiers into a single and hopefully more accurate classifier.