# Monet is spot, Manet is people

Tuzhilina Elena

## 1 Motivation

It takes some time for a human being to conceive the differences between two of the most well-known impressionist painters Claude Monet and Edouard Manet. Although quite cultivated the French beau monde of the nineteenth century had a struggle with this problem as well. When Claude Monet made his debut at the salon in Paris in 1865, his landscapes were displayed next to the famous painting "Olympia" by Edouard Manet who was already known at the time. Funnily enough, people came to Manet to appraise "his" outstanding marinas. Manet was furious: some no-name artist used a similar name to steal his fame. The confusion was overcome and one and a half centuries later the universal source of wisdom, i.e. Internet memes, provides us with the following concise motto: "Monet is spots, Manet is people."

(a) Claude Monet, "Le Déjeuner sur l'herbe" (English: Dinner on the Grass)

(b) Edouard Manet, "Le Déjeuner sur l'herbe" (English: The Luncheon on the Grass)

## 2 Machine Learning problem statement

The problem described above can be restated in terms of Machine Learning and implies three main subproblems:

- **Quantitative data representation**: construct an input space $\mathcal{X} \subseteq \mathbf{R}^d$ and a one-to-one mapping $painting \to x \in \mathcal{X}$. Thus each painting can be represented as a vector in $\mathbf{R}^d$.
- **Dimension reduction**: find an alternative representation of the paintings $x'$ and a new input space $\mathcal{X}' \in \mathbf{R}^{d'}$ where $d' \ll d$ such that each $x \in \mathcal{X}$ can be recovered from $x'$ with high accuracy. In other words, the goal is to find the set of *essential* distinctive features for the paintings.
- **Classification**: construct a Monet vs. Manet classifier, i.e. the hypothesis function $h : \mathcal{X}' \to \mathcal{Y}$, where the output space $\mathcal{Y} = \{"Claude\ Monet", "Edouard\ Manet"\}$ refers to the artists names.

# 3 Data

The raw data used for the research was obtained from "Painter by numbers" kaggle competition (`https://www.kaggle.com/c/painter-by-numbers/data`). Initially, it consisted of 80Gb of images in .png format downloaded from the visual art encyclopedia Wikiart (`www.wikiart.org`) and representing the paintings of about 2000 various artists. Using the data description available on kaggle website, all the painters except Claude Monet and Edouard Manet were automatically filtered leaving the data set of 729 .png images. Although most of the images in the obtained data set were multi-color, there were three Manet sketches in black and white. To make the data homogeneous these three paintings were excluded from the consideration as well. Thus, the final data size after filtration is: 498 paintings by Claude Monet and 228 paintings by Edouard Manet.

# 4 Methodology and tools

## 4.1 Quantitative data representation

First problem here is that all paintings have different resolutions. For the data under consideration the height and width of the painting can vary from 400 to 7500 pixels. To homogenize the data the Python Imaging Library (PIL) is used. First, each image is cropped to be a square with a side equal to the shorter one of the initial rectangle. Second, the already square image is resized to be $300 \times 300$ pixels.

Next, the 8-bit RGB color representation is used to encode the color of each pixel, so the painting representation is a 3-dimensional matrix $300 \times 300 \times 3$ where each matrix element is a number from 0 to 255. For the sake of convenience the 3-dimensional matrix is flattened to a vector of length $2.7 \cdot 10^5$. Therefore, the input space is $\mathcal{X} = \{0, \dots, 255\}^{270000}$ and the dimension of feature matrix $X$ is $726 \times 270000$.

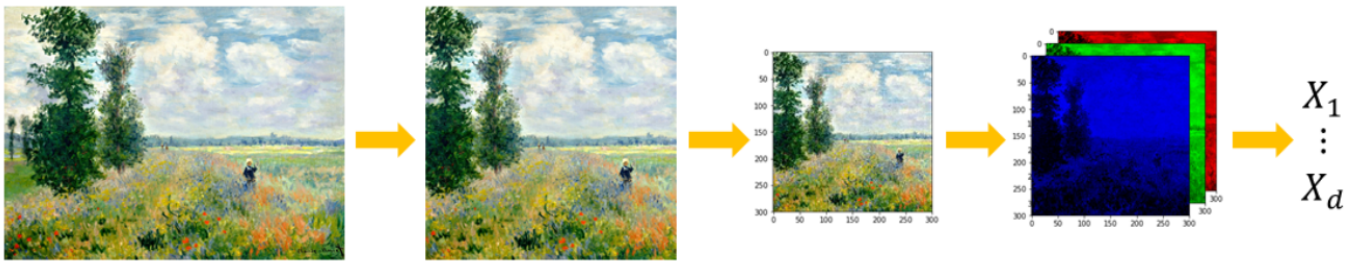



Figure 2: Image preprocessing

## 4.2 Dimension reduction by PCA

Currently, the input space dimension (270000 features) is much higher than the data size (= 726 paintings), that usually implies many problems at the classification step. In addition to the *overfitting* problem common for all classification algorithms, there is the *curse of dimensionality* issue critical for metric algorithms, such as K-Nearest Neighbors.

To reduce the large feature space dimension the *Principal Component Analysis (PCA)* is applied. Recall that centering and scaling transformation is required prior to the application of PCA. Knowing that every current feature is taking value in $\{0, \dots, 255\}$, as the first step, the element wise transformation $f(x) = \frac{x}{255} - 0.5$ is applied to the feature matrix $X$ mapping the range of each element from $[0, 255]$ to $[-0.5, 0.5]$.

Next, the PCA was applied to the transformed data. To have a glance at the "Claude Monet" and "Edouard Manet" classes distribution it is convenient to visualize the data projecting it to the first three principal components space. Note, that in three dimensions it looks very messy and completely non-separable, however, there is a hope that using the higher dimension it would be possible to perform a qualitative classification.

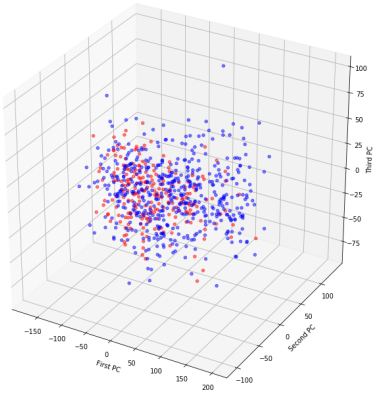



Figure 3: The first three principal components space.

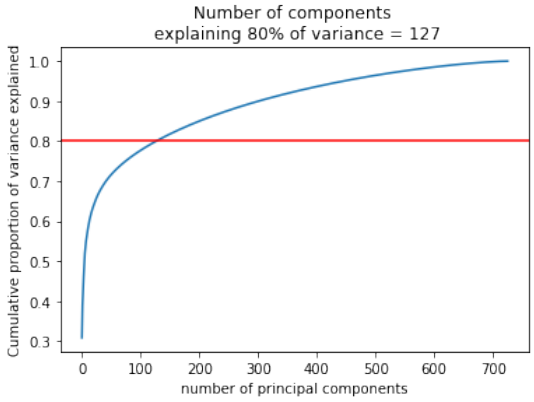
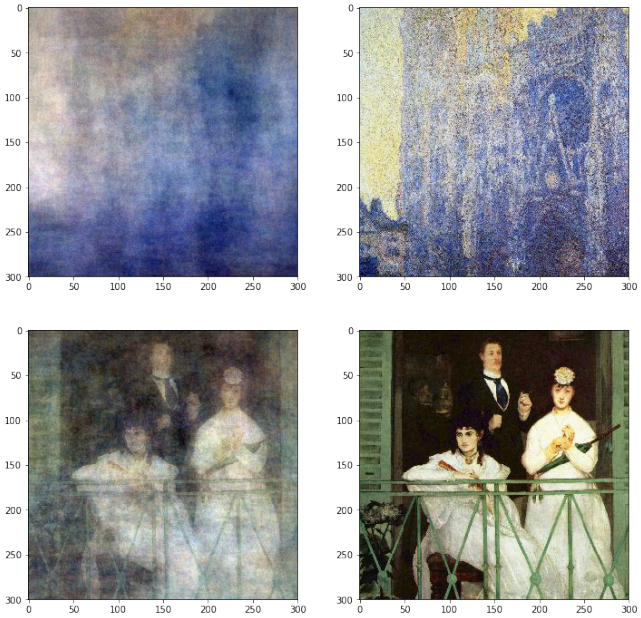


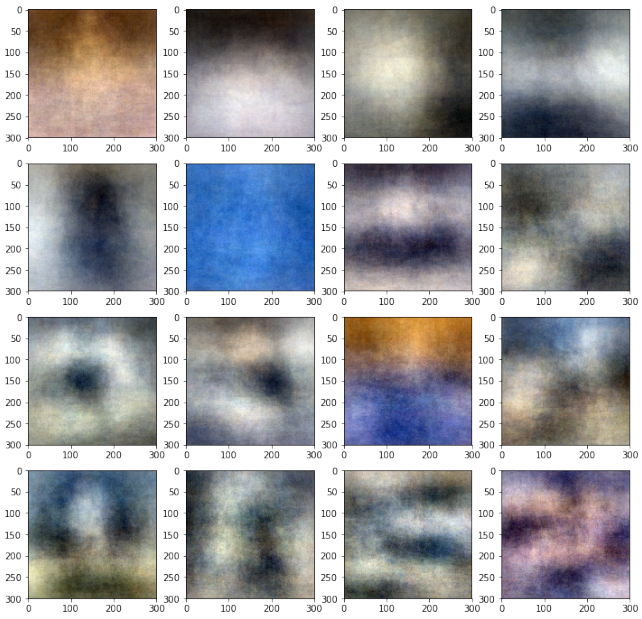

Figure 4: Cumulative proportion of variance explained plot.

To find the appropriate number of principal components the standard variance explained threshold method is used. Plotting the proportion of variance explained vs. the number of components and using the threshold of 0.8 for the proportion of variance explained, we conclude that the first 127 principal components (out of 726) explain 80% of variance, which allows to obtain the significant dimension reduction losing only 20% of the information. Calling the corresponding principal components $v_1, \ldots, v_{127} \in \mathbf{R}^{270000}$ we notice that every painting $x \in \mathcal{X}$ can be approximated by the linear combination of $v_1, \ldots, v_{127}$. Therefore, the found set of 127 principal components can be interpreted as the set of *essential* distinctive features for the paintings. The feature matrix $X'$ corresponding to the new input space $\mathcal{X}' \subseteq \mathbf{R}^{127}$ is obtained as a projection of the initial input space $\mathcal{X} \subseteq \mathbf{R}^{270000}$ onto the linear span of $\{v_1, \ldots, v_{127}\}$ leading to the $726 \times 127$ matrix $X'$.

We conclude the dimension reduction subsection with two visualizations of PCA results. As each painting $x$ in the data set can be approximated by a linear combination of the $v_1, \ldots, v_{127}$ with some coefficients $x'_1, \ldots, x'_{127}$ we plot $x'_1 \cdot v_1 + \ldots + x'_{127} \cdot v_{127}$ to visualize the projection of painting $x$ onto 127−dimensional space spanning the chosen principal components. Note that the first 127 principal components can recover very well the general structures (such as colors and shapes) as well as some minor details (such as umbrella and tie) for both Monet landscape and Manet portrait. Second, we plot the images of the first 16 principal components $v_1, \ldots v_{16}$, which can be interpreted as 16 the most essential paintings features. Interestingly, most of these features can be divided into two general categories: color features (e.g. $v_1, v_6$) and shape features (e.g. $v_7, v_8, v_9, v_{10}, v_{14}, v_{15}$). Moreover, there is one feature $v_{11}$ that is, probably, responsible for the landscape structure meaning that if painting is a landscape with contrast sky and ground, then the corresponding coefficient $x'_{11}$ should be significant in $x'_1 \cdot v_1 + \ldots + x'_{127} \cdot v_{127}$ decomposition.



(a) Claude Monet and Edouard Manet images recovered from the first 127 principal components

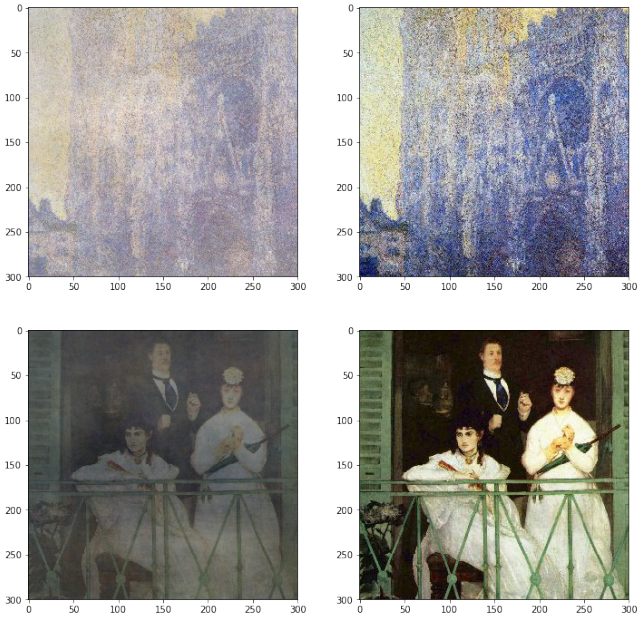


(b) Images of the first 16 principal components

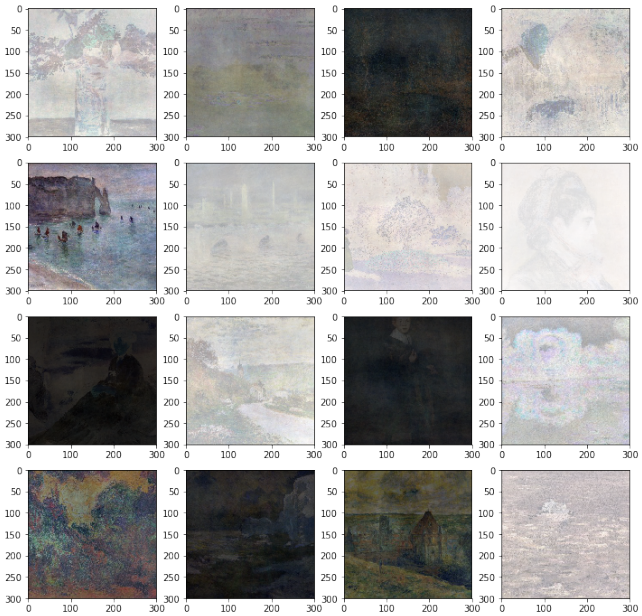
### 4.3 Dimension reduction by ICA

Second dimension reduction method considered in this research is the Independent Component Analysis. The main difference between PCA and ICA is that the first approach restricts the components $v_1, \ldots, v_d$ to be orthogonal, whereas, the goal of ICA is to find the independent(in probability sense) components $v_1, \ldots, v_d$.

For the sake of comparison we plot 16 independent components(filters) and the projections of the same Monet landscape and Manet portrait onto the space spanning independent components. The important observation from both of these plots is that obtained independent components represent the whole painting, possibly, with some minor noise. Therefore, independent components cannot be interpreted as explicit paintings features and so they seems to be useless for further classification purposes.



(a) Claude Monet and Edouard Manet images recovered from independent components



(b) Images of 16 independent components

### 4.4 Classification

As the next step, the classification method is applied to the output space $\mathcal{Y} = \{"Claude\ Monet", "Edouard\ Manet"\}$ and the new input space $\mathcal{X}'$ which the space spanning the first 127 components. Two main approaches were considered: KNN and Kernelized SVM. As both of them require the data standardization, we scale to $[0, 1]$ range each column of the new feature matrix $X'$.

#### 4.4.1 KNN

Comparing the Claude Monet and Edouard Manet paintings it is easy to notice that Monet created many landscapes in green and blue colors, whereas, most of Manet paintings are portraits in dark brown colors. So the first natural idea was to use KNN algorithm to treat the images in one color (which presumably should be the closed points of the feature space in terms of Euclidean distance) as the paintings created by the same author.

Two different parameters for the KNN approach are considered:

- the number of neighbors $k$ considered for each data point
- *weights*, what can be either uniform (all k neighbors are weighted equally) or can equal to the inverse of the Eucledian distance from the neighbor to the point decreasing the importance of the distant neighbors.

#### 4.4.2 SVM

The next classification method under consideration is Kernalized SVM. Three following different kernels were considered in this set up:

- linear kernel $K(x, y) =< x, y >$
- polynomial kernel $K(x, y) = (< x, y > +1)^r$ with parameter $r$
- rbf kernel $K(x, y) = \exp(-\gamma||x - y||^2)$ with parameter $\gamma$,

here $< x, y >$ refers to the inner product of vectors $x$ and $y$.

In addition to the kernel parameters listed above SVM has one common to all kernels parameter $C$ trading off misclassification of training examples against simplicity of the decision surface (the lower the C the smoother the decision surface).

## 5 Results for KNN and Kernelized SVM

### 5.1 Find the optimal parameter

First, the 20% of the data (146 images: 100 by Claude Monet and 46 by Edouard Manet) is held out as a *test set*. The remaining 80% of the data (580 images: 398 by Claude Monet and 182 by Edouard Manet), i.e. the *training set*, is used to find the optimal hyperparameters for each of the classification methods described above. To do so, the grid search with 5-fold cross validation was applied to the training data measuring the mean accuracy on 5 folds for each parameters combination from the grid. Namely, the following grids were considered:

- KNN: $k = \{2, 3, \ldots, 20\}, \quad weights = \{'uniform',\ 'distance'\}$
- linear SVM: $C = \{10^{-3}, 10^{-2}, \ldots, 10^3\}$
- polynomial SVM: $C = \{10^{-3}, 10^{-2}, \ldots, 10^3\}, \quad r = \{2, 3, \ldots 10\}$
- rbf SVM: $C = \{10^{-3}, 10^{-2}, \ldots, 10^3\}, \quad \gamma = \{10^{-5}, 10^{-4}, \ldots, 10^3\}$

After the Grid seach the following optimal hyperparameters and the cross-validation scores were obtained:

- KNN: $k = 2, \quad weights =' uniform', \quad CV - score = 0.748$
- linear SVM: $C = 0.01, \quad CV - score = 0.748$
- polynomial SVM: $C = 0.01, \quad r = 0, \quad CV - score = 0.748$
- rbf SVM: $C = 100, \quad \gamma = 0.0001, \quad CV - score = 0.744$

### 5.2 Measure the algorithm performance

To measure the best algorithm performance with the hyperparameteres listed above the class prediction on test data is made and compared to the true test labels. According to the tables (see Figure 7) the best algorithm in terms of average precision, and f1-score is KNN, whereas, all SVM algorithms perform approximately the same and outperform KNN in terms of recall metric. Note, that for all of the algorithms the proportion of recognized Claude Monet painting is much higher than the proportion of correctly predicted Edouard Manet painting, the most considerable imbalance is observed for the KNN algorithm. This be explained by the fact that there is a class imbalance in the data(498 paintings by Claude Monet and 228 paintings by Edouard Manet) and so the learning algorithm pays more attention to Claude Monet than to Edouard Manet.

| | precision | recall | f1-score | support |
|---|---|---|---|---|
| 0 | 0.24 | 0.85 | 0.37 | 13 |
| 1 | 0.98 | 0.74 | 0.84 | 133 |
| avg / total | 0.91 | 0.75 | 0.80 | 146 |

(a) KNN

| | precision | recall | f1-score | support |
|---|---|---|---|---|
| 0 | 0.59 | 0.68 | 0.63 | 40 |
| 1 | 0.87 | 0.82 | 0.84 | 106 |
| avg / total | 0.79 | 0.78 | 0.79 | 146 |

(b) linear SVM

| | precision | recall | f1-score | support |
|---|---|---|---|---|
| 0 | 0.59 | 0.68 | 0.63 | 40 |
| 1 | 0.87 | 0.82 | 0.84 | 106 |
| avg / total | 0.79 | 0.78 | 0.79 | 146 |

(c) polynomial SVM

| | precision | recall | f1-score | support |
|---|---|---|---|---|
| 0 | 0.61 | 0.67 | 0.64 | 42 |
| 1 | 0.86 | 0.83 | 0.84 | 104 |
| avg / total | 0.79 | 0.78 | 0.78 | 146 |

(d) rbf SVM

Figure 7: Performance of the algorithm with the best hyperparameters on the training set

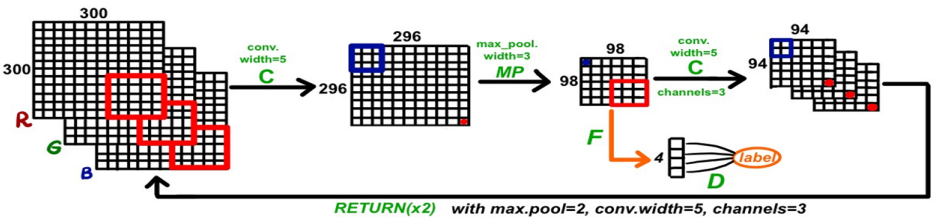
## 6 CNN

To improve the classification results I decided to consider a completely different approach and use Neural Networks. The most common type of Neural Networks used for the purposes of image recognition is Convolution Neural Network. The data representation used in this part of the research is the $300 \times 300 \times 3$ RGB representation obtained after the cropping and rescaling of the original image (see subsection 4.1).

The main ingredients of the model I considered were convolution layers with ReLU activation function(with parameters to be *number of channels*, *size of kernel*), max-pooling(with parameters to be *polling size*) and one fully-connected layer with sigmoid activation function to predict the probability of artist to be Claude Monet(class 1). The goal was to find the balance between the number of information reduced by max-pooling layer, the number of parameters in the final dense layer and the flexibility of the model controlled by the number of channels and the kernel size in the convolution layers.

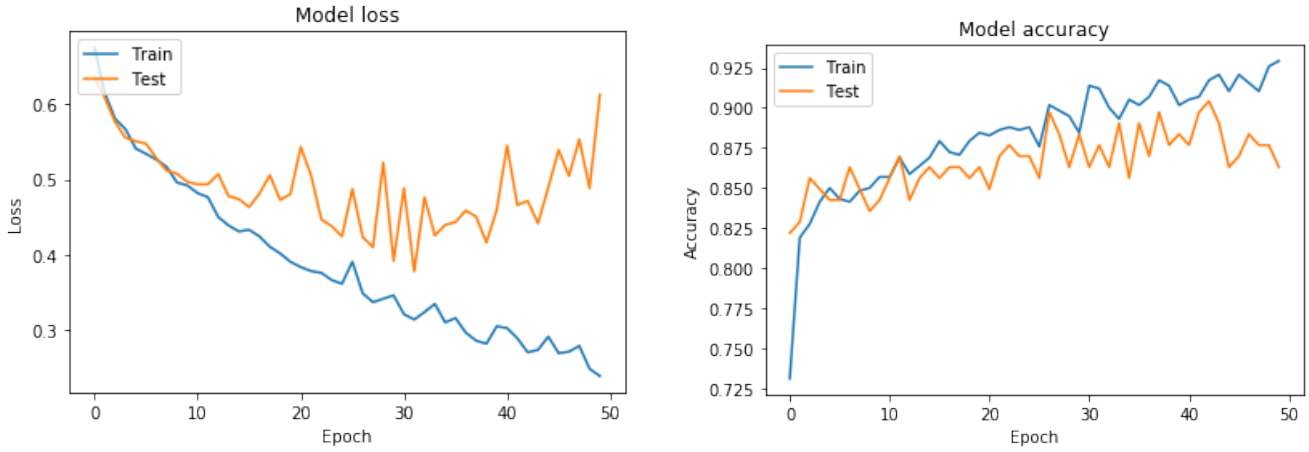
After many experiments I ended up with the following CNN structure:

1. **convolution layer**: kernel size = (5, 5), channels = 1
2. **max-pooling layer**: max-pooling size = (3, 3)
3. **convolution layer**: kernel size = (5, 5), channels = 3
4. **max-pooling layer**: max-pooling size = (2, 2)
5. **convolution layer**: kernel size = (5, 5), channels = 1
6. **max-pooling layer**: max-pooling size = (2, 2)
7. **convolution layer**: kernel size = (5, 5), channels = 3
8. **max-pooling layer**: max-pooling size = (2, 2)
9. **convolution layer**: kernel size = (5, 5), channels = 1
10. **max-pooling layer**: max-pooling size = (2, 2)
11. **flattening +dense layer**: max-pooling size = (2, 2)



The intuition behind the choice of the model parameters is following. First, the number of channels equal to 3 can give the model enough flexibility. Second, max-pooling size equal to (2,2) should, on the one hand, reduce two times the number of parameters used in a dense layer and, on the other hand, make the amount of information lost at max-pooling layer to be not very significant. Finally, the kernel size equal to (5,5) allows to take into account the influence of dilated input elements after several convolution layers with this size of kernel.

The learning parameters were set up to *binary cross-entropy loss*, adam optimization algorithm, batch size equal to 10 and number of epochs equal to 50. In addition to *binary cross-entropy loss* I tracked *accuracy* metric. According to the plots the training loss is decreasing with epochs and the test loss has U-shape which indicates the beginning of an overfitting after $30-$th epoch (see Figure 8).




(a) Binary cross-entropy loss

(b) Accuracy

Figure 8: CNN performance with epochs

## 7 Comparison of the results

The basic metrics calculated for the comparison of KNN, SVM and CNN were *True Positive*, *True Negative*, *Accuracy*, *Area under ROC curve*. Best performance for the models under consideration is:

- TN=0.74(CNN),
- TP=0.98(KNN),
- accuracy=0.86(CNN),
- auroc=0.85(CNN)

According to the plot(see Figure 9), the following conclusions can be made:

- SVM is biased and KNN is very biased towards Monet(class 1), as the TP is much higher than TN for these methods.
- CNN improves *accuracy* and *auroc* by 10%.
- CNN have balanced *TP* and *TN* values and decreases the difference between *TP* and *TN* to 0.18 (comparing to 0.74 for KNN and 0.28 for SVM).

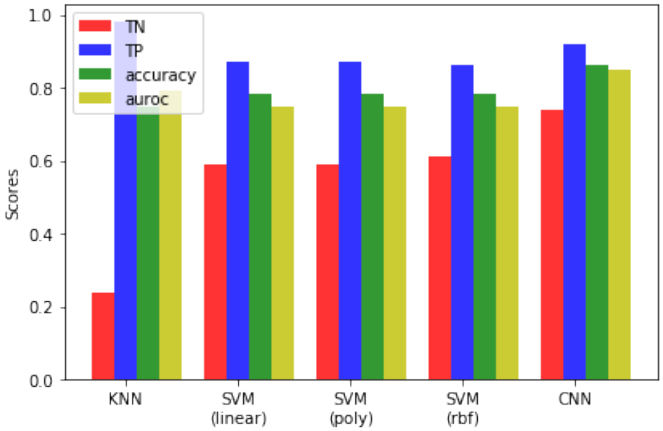



Figure 9: Comparison of methods.
**Conclusion**: CNN can be considered as a better alternative to KNN and SVM.