Cifar-10 Classification with Deep Convolutional Neural Networks and K Nearest Neighbour and Comparison of Them

Oğuzhan Güldamlası
Computer Engineering Department
TOBB University of Economics &Technology
Ankara, Turkey
oguzhanguldamlasi@gmail.com

Buğra Baslı
Computer Engineering Department
TOBB University of Economics &Technology
Ankara, Turkey
bgrbasli@gmail.com

***Abstract*— We tried to predict images in Cifar-10 dataset using artificial intelligence and lazy learner techniques. By using our data on 2 models such as CNN and KNN we tried to obtain the best prediction score. We build our models to classify 10000 test images into 10 different classes. Our work showed us that Convolutional Neural Networks outperforms every preprocessing technique we tried on KNN. To prevent overfitting in CNN we used regularization method called "dropout" and in KNN we tried to extract features from images with color histogram and histogram oriented gradients. Both K-Nearest-Neighbor (KNN) and Convolutional Neural Network (CNN) classification are well known and widely used. We were able to observe that the CNN classifier outperformed the KNN classifier. For future work, we hope to use more categories for the objects and to use more sophisticated classifiers.**

I. Introduction

The human ability to analyze and classify objects and scenes rapidly and accurately is something that everybody finds highly useful in everyday.Humans are able to categorize complex natural scenes very quickly . In order to understand a complex scene, the first step is to recognize the objects and then recognize the category of the scene. In order to do this we use Convolutional Neural Networks and K nearest neighbor classifiers that all have different characteristics and features.

Various classifiers are used in algorithms that involve object recognition. However object recognition is challenging for several reasons. The first and most obvious reason is that there are about 10,000 to 30,000 different object categories. The second reason is the viewpoint variation where many objects can look different from different angles. The third reason is illumination in which lighting makes the same objects look like different objects. The fourth reason is background clutter in which the classifier cannot distinguish the object from its background. Other challenges include scale, deformation, occlusion, and intra-class variation. Applications for classification in computer vision include computational photography, security, surveillance, and assistive driving.

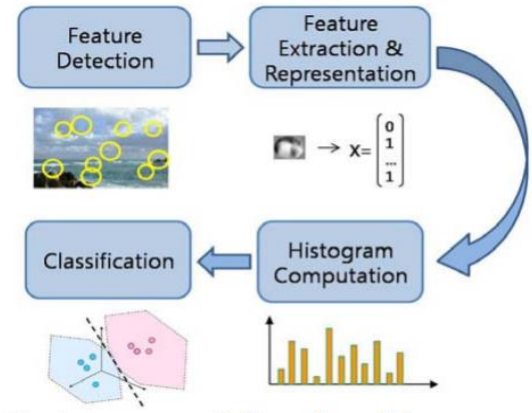

Figure 1.Summary of classifcation of an image in Artifical Intelligence

Current approaches to object recognition make use of machine learning methods. To improve performance , we can get larget datasets, learn powerful models and use better techniques for prevent overfitting.To learn about 10 class of objects from 50000 images , we need a model with a large learning capacity and good preprocessing techniques.CNNs make strong and mostly correct assumptions about nature of images but KNN only calculates distances to nearest images.Thus , in most cases CNN outperformed KNN algorithm. Despite the attractive qualities of CNNs , they have still been prohibitevly expensive to apply to images. But current GPUs(We trained on GeForce 750m and GeForce GTX 1050TI) are powerful enough to perform 2D convolution and feature extraction.

The contributions of this article as follows : we trained a large convolutional neural network and K nearest neighbour and we compared the results of algorithms with different preprocessing methods and parameters. In CNN our network contains a few features which improve its performance which are detailed later in this paper. Also in KNN our algorithm has a few feature which effects our prediction score on test images.

The overfitting and curse of dimensionality was our greatest challange in this project.So prevent overfitting in CNN we used "dropout" (according to [1] for brief Dropout explanation) technique and in KNN we extract features from images with color histograms and histogram oriented gradients.

In the end we compared our results with different algorithms and parameters.Network's size is limited becaues of memory and time.And KNN was very slow to predict images while calculating distances to 50000 images.

II. The Dataset

The CIFAR-10 are labeled subset of 80 million tiny images dataset.And collected by Alex Krizhevsky,Vinod Nair and Geoffrey Hinton

The CIFAR-10 dataset consists of 60000 32x32 images in 10 classes ,with 6000 image per class.There are 50000 training images and 10000 test images.

The dataset is divided into five training batches and one test batch , each with 10000 images.The test batch contains 1000 randomly selected images from each class . The training batches contain the remaining images in random order.And the training batches contain exactly 5000 images from each class.

Here are the classes in the dataset and 10 random images from each class.

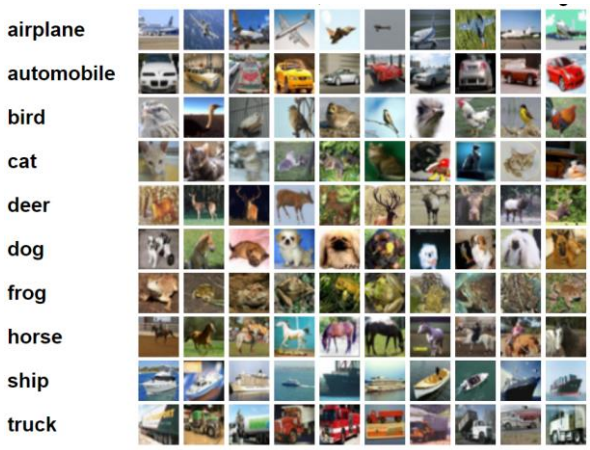

Figure 2. Summary of CIFAR-10 dataset

The classes are mutually exclusive . For example there is no overlap between automobiles and trucks."Automobile" includes sedans,SUVs etc."Truck" includes only big trucks.

A batch is 10000x3072 array of uint8s. Each row of the array stores a 32x32 colour image. The first 1024 entries contain the red channel values, the next 1024 the green, and the final 1024 the blue. The image is stored in row-major order, so that the first 32 entries of the array are the red channel values of the first row of the image.

Labels are a list of 10000 numbers in the range 0-9. The number at index *i* indicates the label of the *i*th image in the array data.

III. Methodology

Even though there are a lot of steps in creating a solution,These steps are generally interwined.We will be separating the methodology into two main parts : Convolutional Neural Networks and K nearest neighbor.Also we will give information about data preprocessing techniques,feature extraction methods we used in project.

A. Methodology of Convolutional Neural Networks and architecture of network

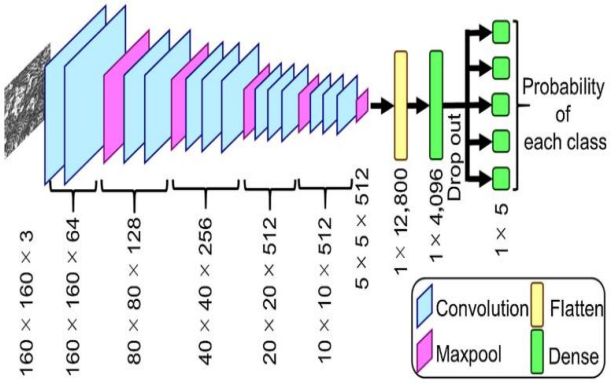

Figure 3. An example of CNN structure.

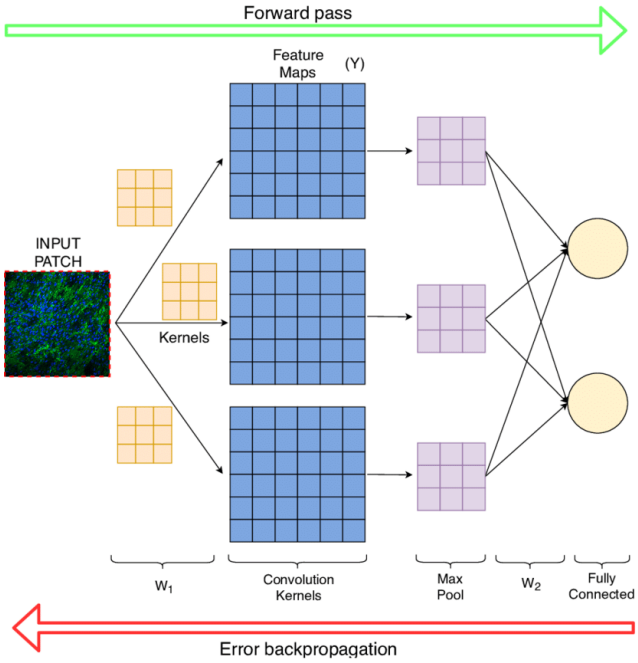

Figure 4.An illustration for CNN(Predicting input,and calculating weights)

First we need to do preprocessing.The original one batch data is (10000x3072) matrix . The number of columns (10000) , indicates the number of sample data. As stated before the row vector , (3072) represents an color image of 32x32 pixels.Since we use CNN for classification task,the original row vector is not appropriate . In order to feed an image data into a CNN model , the dimension should be width x width x num_channel.

The problem with CNN was the complexity of model.A model with more layer becomes complex model for our data on the other hand a model with less layer becomes simple model. We tried the model with fixed hyper parameters and functions to find best model for our data.

Later we need to normalize the data between 0 and 1 (inclusive) for this we convert our images to float then divided by 255.By applying this the original image data transformed in to range 0 to 1. We did this because we used

ReLU activation function , and if input value somewhat large, the output value increase linearly so the output could be very huge so when back-propagation process is performed to optimize the networks , this could lead to an vanishing gradient problem. Thus, we normalized our data.

Our output of model is a set of probabilities of each class of image based on model's prediction result.In order to express those probabilities , a vector having the same number of elements as the number of classes of the image is needed.CIFAR-10 provides 10 different classes of the image, so we need a vector in size of 10 as well. For CIFAR-10 dataset there is one-hot encoding process.

index	label
0	airplane (0)
1	automobile (1)
2	bird (2)
3	cat (3)
4	deer (4)
5	dog (5)
6	frog (6)
7	horse (7)
8	ship (8)
9	truck (9)
...	...
...	...

label	index											
	0	1	2	3	4	5	6	7	8	9	...	...
airplane	1	0	0	0	0	0	0	0	0	0	...	...
automobile	0	1	0	0	0	0	0	0	0	0	...	...
bird	0	0	1	0	0	0	0	0	0	0	...	...
cat	0	0	0	1	0	0	0	0	0	0	...	...
deer	0	0	0	0	1	0	0	0	0	0	...	...
dog	0	0	0	0	0	1	0	0	0	0	...	...
frog	0	0	0	0	0	0	1	0	0	0	...	...
horse	0	0	0	0	0	0	0	1	0	0	...	...
ship	0	0	0	0	0	0	0	0	1	0	...	...
truck	0	0	0	0	0	0	0	0	0	1	...	...

Figure 5. Shows the one-hot-encoding process.

After the preprocessing we began the construct our neural network.First we try to choose our functions(activation,loss,optimizer).According to our researches we used ReLU Nonlinearity for activation function according to Alex Krizhevsky,Ilya Sutskever,Geoffrey Hinton[2] in their paper and there is a brief explanation why ReLU outperforms other activation functions in CNNs.

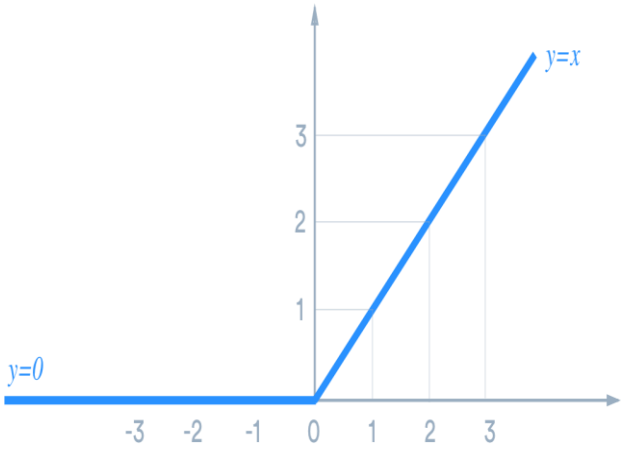

Figure 6. ReLU Activation Function.

Then we picked our main optimizer function SGD (Stochastic Gradient Descent).But we tried a few optimizer functions to see results(RMSPROP).

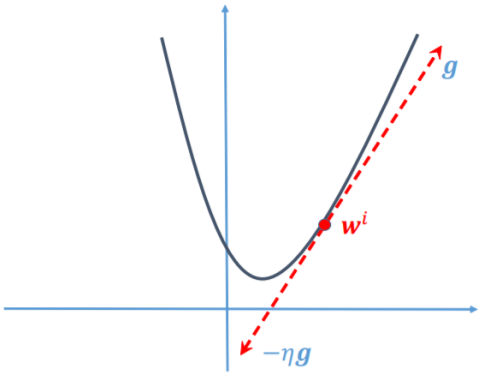

Figure 7.Illustration of the gradient descent method.Where n is learing rate.

Finaly we choose Categorical Cross Entropy for loss function.

$$-\sum_{c=1}^{M} y_{o,c} \log(p_{o,c})$$

Figure 8.Cross Entropy Function.

After that we started to build our model with convolutional layers and neural networks.In our model there is 3 convolutional layers and 2 fully connected layers.

First convolutional layer is Convolution with 32 different filters in size of (3x3) Max Pooling by 2 with ReLU activation function

Second convolutional layer Convolution with 64 different filters in size of (3x3) Max Pooling by 2 with ReLU activation function

Third convolutional layer Convolution with 128 different filters in size of (3x3) Max Pooling by 2 with ReLU activation function

After convolution operations the layers as follows:

1)Flattening the 3-D output of the last convolving operations.

2)Fully Connected Layer with 128 units(ReLU activation)

4)Fully Connected Output Layer with 10 units (number of classes).

And there is a dropout after each hidden layer in network with probability 0.2 to prevent overfitting.Dropped neurons will not be used in forward pass and backpropagation (for one sample).

We tried different models, functions and parameters.See results section for their performances.

In CNN our best accuracy was %82.44 with these parameters and functions:

Parameter	Value
Batch size	64
Number of epochs	100
Learning Rate	0.001
Momentum	0.9
Optimizer	SGD
Activation Function	ReLU
Loss Function	Categorical Cross Entropy
Dropout Probability	0.3

Figure 9.Shows the best score with parameters

B. *Methodology of K nearest neighbor*

K nearest neighbor is a lazy learner algorithm so there is not a training part.

K nearest neighbor algorithm is a method for classifying objects based on closest training examples in the feature space. k-nearest neighbor algorithm is among the simplest of all machine learning algorithms. Training process for this algorithm only consists of storing feature vectors and labels of the training images. In the classification process, the unlabelled query point is simply assigned to the label of its k nearest neighbors.

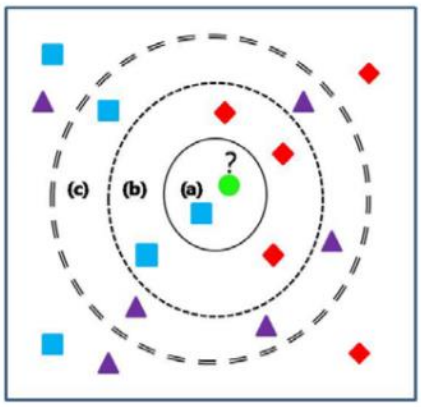

Figure 10. KNN classification.

Typically the object is classified based on the labels of its k nearest neighbors by majority vote. If k=1, the object is simply classified as the class of the object nearest to it. When there are only two classes, k must be a odd integer. However, there can still be ties when k is an odd integer when performing multiclass classification. After we convert each image to a vector of fixed-length with real numbers, we used the most common distance function for KNN which is Euclidean distance:

$$d(x,y) = \|x-y\| = \sqrt{(x-y)\cdot(x-y)}$$

Figure 11. Euclidean distance formula

There is a really important problem with KNN that is curse of dimensionality. Also after a certain value of K model's accuracy decreasing because of high bias and before a certain value of K accuracy is decreasing because of high variance. As long as we consider these informations we tried different preprocessing techniques.

Giving exact image as an input is a bad choice because of curse of dimensionality. Our feature vector's size for one image is 3072(32x32x3) without any feature extraction.

First we tried our model without any preprocessing method. Our result was not enough to classify other images (approximately 0.335-0.316).Then we searched ways to extract features from images. We saw that extracting features from color histograms and histogram oriented gradients is useful for our case and popular. Thus, we tried that 2 feature extraction methods for our K nearest neighbor algorithm.

After the search , first we tried color histogram method. In this method we extract the histogram of each image with 8 color levels, then we calculate a input vector from these histograms using their 8 values(8x8x8=512 features) and normalized them in range 0 and 1.After the predictions of 10000 images we get accuracies in range 0.31-0.34.These results were slightly better than previous ones.

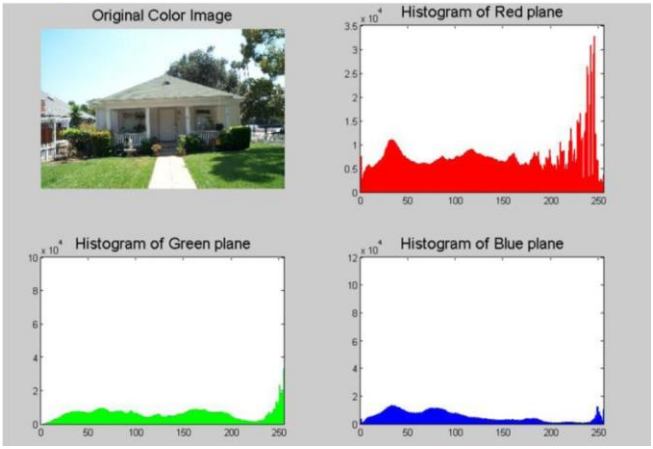

Figure 12. Color Histogram for an input image

Then we saw that there is a overfitting problem caused by the curse of dimensionality. We applied histogram oriented gradients [3]. Method reduced our feature size to 324 . We tried to predict our images with these 324 features. After that our results was in range 0.51 – 0.56.That is a great improvement for our KNN algorithm.

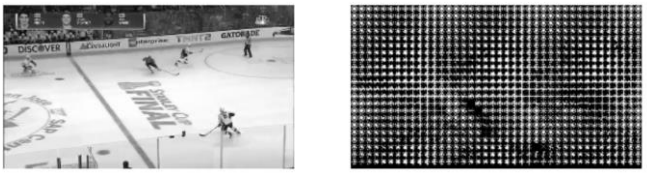

Figure 13. Histogram oriented Gradients of input image.

IV. RESULTS

We did different preprocessing techniques for KNN algorithm and we tried different parameters and functions for CNN algorithm. Nearly every time we get different accuracy scores from our models depending on parameters , functions etc.

A. Results for convolutional neural networks

Optimizer Function	Batch size	Number of epochs	Model Layers	Learning Rate	Accuracy(%)
stochastic gradient descent(without Dropout)	128	100	3 Convolutional Layers 2 Fully Connected Layer	0.9	45
stochastic gradient descent(with Dropout)	64	100	3 Convolutional Layers 2 Fully Connected Layer	0.6	82+
RMSPROP (without Dropout)	128	100	3 Convolutional Layers 2 Fully Connected Layer	0.9	68
RMSPROP(with Dropout)	64	100	3 Convolutional Layers 2 Fully Connected Layer	0.6	73+
RMSPROP (with Dropout)	64	100	2 Convolutional Layers 2 Fully Connected Layer	0.6	69+

Figure 14.CNN accuracy results for different parameters,layers,functions.(Momentum value for all results is 0.9)

As can be seen overfitting prevent techniques , model complexity and different hyper parameters are important on test accuracy. Also we saw that dropout technique is important to use to avoid overfitting and setting learning rate depending on batch size is important.Also simple models have less accuracy because of high bias. Finally we can say that using techniques like dropout etc. and setting bias-variance tradeoff is really affects accuracy in Artifical Intelligence problems.

B. Results for K nearest neighbor

K values	Feature Extraction Technique	Accuracies respectively to k values
1,10,30,50,100	Without any feature extraction (3072 features)	K:1 => 0.338 K:10 => 0.345 K:30 => 0.340 K:50 => 0.333 K:100 => 0.317
1,10,30,50,100	Extraction from 8x8x8 color histogram(512 features)	K:1 => 0.3084 K:10 => 0.365 K:30 => 0.355 K:50 => 0.346 K:100 => 0.339
1,10,30,50,100	Extraction from histogram oriented Gradients(324 features)	K:1 => 0.51 K:10 => 0.55 K:30 => 0.5449 K:50 => 0.5442 K:100 => 0.5235

Figure 15. KNN accuracy results for different feature extractions and K values.

As can be seen optimal value of K for our problem is bigger than 10 and smaller than 30. Depending on the K value model's bias and variance varies.So it have to be set well. And curse of dimensionality is a challenging problem for KNN algorithm. We tried 2 solutions to combat to this problem and one of them (hog method) was very effective and it can be seen on the Figure 15.

A main advantage of the KNN algorithm is that it performs well with multi classes because the basis of its decision is based on a small neighborhood of similar objects. Therefore, even if the target class is multi-modal, the algorithm can still lead to good accuracy. However a major disadvantage of the KNN algorithm is that it uses all the features equally in computing for similarities. This can lead to classification errors, especially when there is only a small subset of features that are useful for classification.

V. CONCLUSION

In this project , we implemented two different classifiers for image classification. We could observe that at most cases CNN outperformed KNN classifcation. Although the performance of KNN was very low compared to CNN's .Also CNN's feature extraction is better than feature extraction methods we used in KNN. With better feature extraction methods and data preprocessing techniques KNN can outperform the CNN but it is hard to do.

In KNN, output completely relies on nearest neighbors, which may or may not be good choice. Also it is sensitive to distance metrics.

On the other hand, CNN extract the features from the input data. Which are very helpful for making analysis.

So we can say that in our dataset CNN is useful to predict image labels.

The performance of various classification methods still depend greatly on the general characteristics of the data to be classified. The exact relationship between the data to be classified and the performance of various classification methods still remains to be discovered. Thus , there has been no classification method that works best on any given problem. There have been various problems to the current classification methods we use today. To determine the best classification method for a certain dataset we still use trial and error to find the best performance. For future work, we can use more different kinds of categories that would be difficult for the computer to classify and compare more sophisticated classifiers. Another area for research would be to find certain characteristics in various image categories that make one classification method better than another.

ACKNOWLEDGMENT

We would like to thank Associate Prof. Dr. Murat Özbayoğlu and Uğur Şahin (master degree in computer science) for supervising us in our project, for useful discussions and for taking the time to us.

REFERENCES

[1] Alex Krizhevsky, Ilya Sutskever, and Geoffrey E. Hinton,Nitish Srivastava,Ruslan Salakhutdnov."Dropout : a simple way to prevent neural networks from overfitting." The Journal of Machine Learning Research January 2014.

[2] Alex Krizhevsky, Ilya Sutskever, and Geoffrey E. Hinton. "Imagenet classification with deep convolutional neural networks." In Advances in Neural Information Processing Systems, pp. 1097-1105. 2012.J. Clerk Maxwell, A Treatise on Electricity and Magnetism, 3rd ed., vol. 2. Oxford: Clarendon, 1892, pp.68–73.

[3] Takumi Kobayashi; Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition (CVPR), 2013, pp. 747-75