Comparison of various models for natural language processing and image classification

Authors: Myriam Laiymani (20140876), Adel Nabli (20121744), Pierr Rosin (20025653),
Lawrence Abdulnour (20019894) and Jean-Hughes Fournier-Lupien (Polytechnique: 1433182)

Introduction

For this study, we propose to compare various models for to specific applications such as natural language processing and image classification. We chose to explore sentiment analysis on the IMDB movie review dataset which consists to predict if a review is positive or negative according to the movie review. Then, we propose to compare different algorithms in order to classify the well-known CIFAR-10 dataset.

Datasets

- IMDB movie review:
  - 25,000 train, 25,000 test
  - Problem: binary classification
- CIFAR-10:
  - 60,000 images (32×32) pixel,
  - 10 categories
  - Problem: multiclass classification

The ToMATo Algorithm

Algorithm 6.1: ToMATo

Input: graph G with n vertices, n-dimensional vector $\tilde{f}$, parameter $\tau \geq 0$.

Sort the vertex indices $\{1, 2, \cdots, n\}$ so that $\tilde{f}(1) \geq \tilde{f}(2) \geq \cdots \geq \tilde{f}(n)$;
Initialize a union-find data structure $\mathcal{U}$ and two vectors g, r of size n;
for $i = 1$ *to* n **do**
  Let $\mathcal{N}$ be the set of neighbors of i in G that have indices lower than i;
  if $\mathcal{N} = \emptyset$ **then** // vertex i is a peak of $\tilde{f}$ within G
    Create a new entry e in $\mathcal{U}$ and attach vertex i to it;
    $r(e) \leftarrow i$; // $r(e)$ stores the root vertex of entry e
  else // vertex i is not a peak of $\tilde{f}$ within G
    $g(i) \leftarrow \arg\max_{j \in \mathcal{N}} \tilde{f}(j)$; // g stores pseudo-gradient edges
    $e_i \leftarrow \mathcal{U}$.find$(g(i))$;
    Attach vertex i to the entry e_i;
    for $j \in \mathcal{N}$ **do**
      $e \leftarrow \mathcal{U}$.find(j);
      if $e \neq e_i$ and $\min\{\tilde{f}(r(e)),\ \tilde{f}(r(e_i))\} < \tilde{f}(i) + \tau$ **then**
        $\mathcal{U}$.union$(e,\ e_i)$;
        $r(e \cup e_i) \leftarrow \arg\max_{\{r(e),\ r(e_i)\}} \tilde{f}$;
        $e_i \leftarrow e \cup e_i$;
      end
    end
  end
end

Output: the collection of entries e of $\mathcal{U}$ such that $\tilde{f}(r(e)) \geq \tau$.

Conclusion

Dataset	ToMATo	SVM	ConvNet
IMDB	77%	89%	90%
CIFAR-10	-	41%	83%

Table 1: Comparison of the accuracy of the different algorithms on the test set

References

[1] Simonyan K., Zisserman A. (2015), ICLR 2015 Conference Paper.1-14

[2] McInnes, L., Healy, J. (2018). arXiv preprint arXiv:1802.03426.

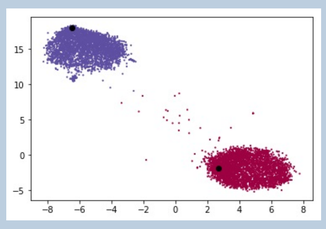
[3] Maaten, Laurens van der, and Geoffrey Hinton, Journal of machine learning research 9.Nov (2008): 2579-2605.

[4] Chen, Y., Li, J., Xiao, H., Jin, X., Yan, S., Feng, J. (2017). In Advances in Neural Information Processing Systems (pp. 4467-4475).

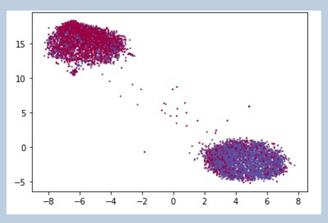
[5] Graham B. Fractional Max-Pooling, (2017)

[6] Oudot, S. Y. (2015). Persistence theory: from quiver representations to data analysis (Vol. 209). Providence, RI: American Mathematical Society.

Natural Language Processing on movie reviews (IMDB)



ToMATo clusters of the "positive" and "negative" reviews.



Comparison with UMAP projection of the data in a 2-D

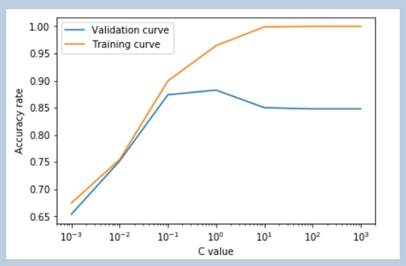
- **Algo: Topological Mode Analysis Tools (ToMATo) applied on IMDB**
- Extraction of adjectives of each reviews using the *spacy* library.
- Preprocessing using: UMAP [2] and ToMATo [6] algorithm.
- ToMATo is better to distinguish the two classes. The clustering algorithm leads to an accuracy 77.4%.

- **Algo: Convolutional Neural Network (CNN) applied on IMDB dataset**
- Preprocessing: embedding layer using Keras library
- Accuracy: 90% on the test set

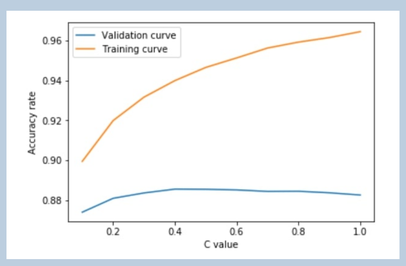
Layer (type)	Output Shape	Param #
embedding_5 (Embedding)	(None, 500, 64)	320000
conv1d_5 (Conv1D)	(None, 500, 64)	24640
max_pooling1d_5 (MaxPooling1	(None, 250, 64)	0
flatten_5 (Flatten)	(None, 16000)	0
dense_9 (Dense)	(None, 250)	4000250
dense_10 (Dense)	(None, 1)	251

Total params: 4,345,141
Trainable params: 4,345,141
Non-trainable params: 0

Architecture of the CNN



Accuracy for different values of penalty parameter C



Accuracy for C value around the optimal value 0.4

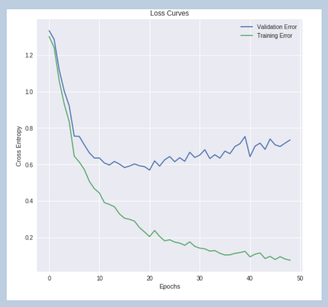
- **Algo: Support Vector Machine (SVM)**
- Preprocessing: bag of words model
- Optimal parameters: linear kernel, $C = 0.4$
- Accuracy: 88.5% on the test set with

Image classification of CIFAR-10:

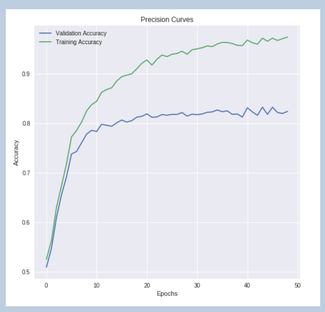
Layer	Kernel	Filters	Stride	Padding	Output Size
ConV/ReLU MaxPool	3x3 2x2	64	1	1	64x16x16
ConV/ReLU MaxPool	3x3 2x2	128	1	1	128x8x8
ConV/ReLU MaxPool	3x3 -	256	1	1	256x8x8
ConV/ReLU MaxPool	3x3 2x2	256	1	1	256x4x4
ConV/ReLU MaxPool	3x3 -	512	1	1	512x4x4
ConV/ReLU MaxPool	3x3 2x2	512	1	1	512x2x2
ConV/ReLU MaxPool	3x3 -	512	1	1	512x2x2
ConV/ReLU MaxPool	3x3 2x2	512	1	1	512x1x1
Fully Connected Layer	-	-	-	-	512x10*

CNN architecture

- **Algo: Convolution Neural Network**
- Preprocessing: data augmentation: random transformations (rotations, and flips) and normalisation of the images (mean = 0, devided by std dev.)
- Architecture inspired by the state-of-the-art architecture adopted by Simonyan and Zisserman [1]



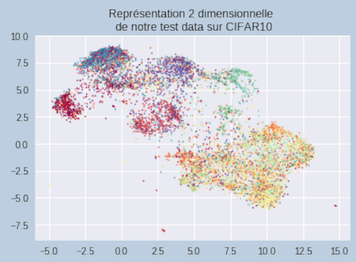
Loss curves (CNN)



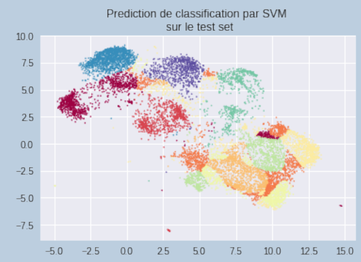
Accuracy curves (CNN)

- Optimal parameters: 50 epoch, learning rate 0.005 and batch size: 64
- Accuracy of 83% on the test set

- **Algo: Support Vector Machine**
- Preprocessing: reshape images to vectors and normalisation of the vector (mean = 0, divided by std. dev.)
- Dimension is reduced to 2 using UMAP algorithm.
- Optimal parameters: linear kernel, $C = 1$ and 800 iterations.
- Accuracy: 41% on the test set.

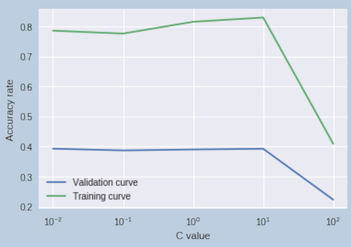


2-D represention of the test set after UMAP algorithm, colorcoded with the labels.



2-D represention of the test set after UMAP algorithm, colorcoded with the predictions of the SVM algorithm

Optimization of the number of iterations with $C = 1$



Optimization of the penalty parameter C with 800 iterations