Landmark recognition using convolution- and transformer-based models

Tájfelismerés konvolúciós és transzformer alapú modellekkel

Adrián Tibor Bodai
Budapest University of Technology and Economics
bodai.adrian.tibor@gmail.com

Judit Hermán
Budapest University of Technology and Economics
juditka010326@gmail.com

Kíra Diána Kovács
Budapest University of Technology and Economics
kira20020111@gmail.com

Kivonat

A gépi látás jelenlegi helyzetét a képosztályozási feladatokban megkövetelt pontosság fokozódó igénye jellemzi, ezzel foglalkozik a Google Landmark Recognition 2021 Kaggle verseny. Ez a feladat a képfelismerés nehézségeit járja körül a gépi tanulás szemszögéből, feltárva a módszertani hátteret és azon kísérleteket, amelyeket a feladat megoldására tettek. A verseny megoldásán keresztül részletesen kifejtük a klasszikus konvolúciós neurális hálózatok (CNN) és a transzformer architektúrák működését. A hagyományos CNN-eken kívül transzformer architektúrákat is alkalmazunk, ami eredetileg természetes nyelvfeldolgozáshoz lett tervezve.

A feladat megoldása közben számítási korlátokba ütköztünk, különösen a Transformer alapú hálózat esetén. Ezek a korlátok, habár akadályozták a terveink megvalósítását, mégis kulcsfontosságú betekintést nyújtottak a transzformer architektúrák használatába.

Abstract

The contemporary landscape of computer vision is marked by an escalating demand for precision in image classification tasks, notably exemplified by the formidable challenge posed in the Google Landmark Recognition 2021 Kaggle competition. This task navigates the difficulties of landmark recognition through the lens of machine learning, unraveling the methodological underpinnings and experimental endeavors undertaken to address the task at hand. Through the solutions of the comptetition we introduce the classical convolutional neural networks (CNNs) and the transformer architectures. Besides of the CNNs, we use transformer architectures, that is originally conceived for natural language processing.

We faced the challenges of computational constraints, most notably evidenced in our Transformer-based network. These limitations, albeit impeding the realization of a model commensurate with our ambitions, instigated crucial insights into the prospective efficacy of transformer architectures.

I. INTRODUCTION, OVERVIEW OF THE SUBJECT AREA, PREVIOUS SOLUTIONS

Our work is based on a Kaggle competition, named Google Landmark Recognition 2021. The competition is about writing a code of a model that can predict landmarks based on some pictures of them. There are about 81 000 classes, alias landmarks, and there are an average of 20 pictures in each class. We trained models (we'll describe them later in this documentation), and try to recognize which landmarks are depicted them. There are some pictures, that are about neither of the landmarks, we need to handle these pictures too (for example with a plus, empty class). We have a training set of 1.6 Million pictures and a .CSV file with the landmarks' labels. Each image has a unique identification number, we can reference to them with it in the code.

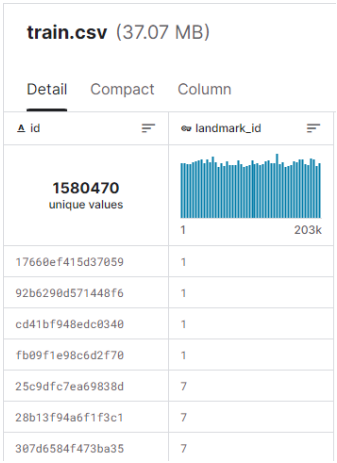

Fig. 1. The train.csv file with the id's and labels, source: Kaggle

Since this competition is over, we could look at the previous solutions of the competitors. The winner of the competition ensembled various models, such as DOLG-Efficientnet b5-b7, Hybrid SwinBase224-Efficientnet b3, b5-b6, and All Data Ext 2020 Efficientnet b3, b6. These are convolutional neural networks (CNNs), which is a type of artificial neural network designed specifically for tasks involving images and visual data. CNNs have been extensively used for object detection, it is a type of feed-forward neural network and works on principle of weight sharing. Convolution is an integration showing how one function overlaps with other function and is a blend of two functions being multiplied. [1] A convincing gain of CNNs over another traditional method is its capability to simultaneously perform following tasks like features extraction, reducing data dimension, and classification in the particular organize network structure. [2]

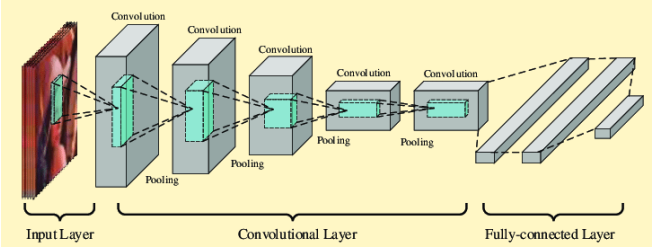

Fig. 2. Convolutional neural network pattern [3]

In recent years, the field of computer vision has undergone a revolutionary transformation with the introduction of transformer architectures. Originally designed for natural language processing tasks, transformers, as outlined in the groundbreaking document "Attention is All You Need," have proven to be exceptionally effective in various domains, including image classification.

The traditional CNNs dominated the landscape of image classification for a long time. However, as the complexity of tasks and the scale of datasets increased, transformers emerged as a versatile alternative. Unlike CNNs, transformers rely on a mechanism called self-attention to capture long-range dependencies in the data, making them particularly well-suited for tasks with intricate relationships between different image elements.

We tried to implement a Transformer based network. Sadly the requirements of our computers were not good enough to make a usable model, but we will write about it a little in our paper, because we had a lot of work in it and if we had a better computer, we could maybe make a much better model with the program.

II. System plan

We used this architecture to create the CNN. This architecture ensured a good pre trained model, and made the training much easier, because we did not have to initiate the first Convolutional layers. First we cut the last Dense layer of the RESNET50 IMagenet, than we added the new Dense layer, containing n neurons. Of course, this n number depended on how many classes we were teaching. In the case of the above picture, we trained on 1971 classes, in the other case we trained on 50.

```
Model: "sequential"
_________________________________________________________________
 Layer (type)                Output Shape              Param #
=================================================================
 resnet50 (Functional)       (None, 4, 4, 2048)        23587712

 flatten (Flatten)           (None, 32768)             0

 dropout (Dropout)           (None, 32768)             0

 dense (Dense)               (None, 1971)              64587699

=================================================================
Total params: 88,175,411
Trainable params: 64,587,699
Non-trainable params: 23,587,712
_________________________________________________________________
```

Fig. 3. The used model's parameters for 1971 classes

III. Data Acquisition and Preparation

We downloaded the data from Kaggle, and due to the nature of the task, there was no need for data cleaning. The competition involved 105 GB of images, spanning over 80,000 categorical classifications. The classes were quite imbalanced, with some having only 2 images while others had over 6000. To address this, we considered augmenting images for classes with a small number of elements. However, due to storage and memory constraints, we filtered out classes with fewer than 100 training images instead. The dataset consists of relatively high-resolution color images, and pictures belonging to different classes often exhibit significant variations. To handle the large data size during minibatch training without overwhelming the memory, we implemented the DataGenerator class, inspired by a solution from another Kaggle user. This class loads the images batch by batch through the DataGenerator call instead of loading the entire dataset into the memory at once, because it would require more than 30 GB of RAM, which we do not possess.

Fig. 4. Sample pictures of the dataset

IV. Training

We conducted training on our local machine with a Laptop GPU RTX3060 . For models uti lizing ResNet50 [7][8][9][10], we initially trained only the l ast layers for 10-15 epochs. This allowed the network to learn task-specific features without compromising the original weights. Subsequently, we made all layers trainable and continued training for an additional 10-15 epochs, further reducing the loss function. Additionally, we explored the possibility of incorporating a transformer.

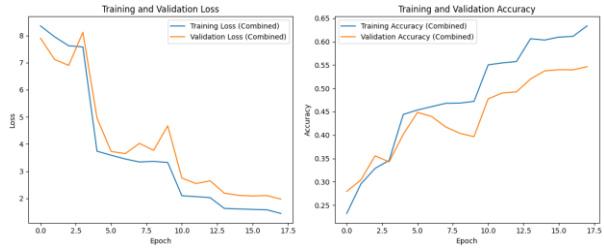

Fig. 5. Training and Validation Loss and Accuracy

As we can see on the above picture, the training loss always goes down, and the training accuracy always goes up. But our goal was not to overfit the model, but to make it capable of generalization. This is the orange line, the validation dataset. We can see, that it fluctuates a lot. Whenever it goes up, the learning rate is reduced (multiplied by 0.4, we tried other values,but this was the most successful), so the model can learn new ways of understanding the picture.

The training was stopped, when there was no improvement in the validation data for 3 whole epochs, to avoid overfitting.

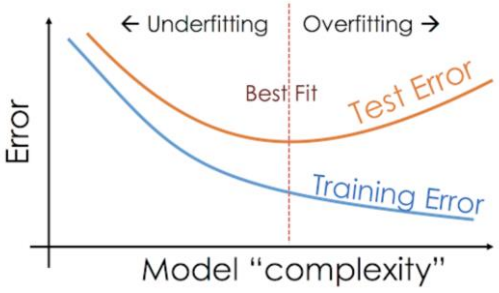

Fig. 6. Over- and underfitting of a model

The other big model was the ResNet50 based 1971 class model. Here is its training process:

Fig. 7. Training process of the ResNet50 based model on 1971 classes

Here we can also see the learning rate for the 17 epochs of the training. The image shows the significance of the EarlyStoppingTime as well, because after this point in the training, the model would become overfitted and its accuracy on the validation dataset would become much worse. Of course, we stopped at the right time, to reach almost 10% accuracy on almost 2000 classes, which is not an insignificant value. We will incorporate it further in the evaluation segment of the paper.

Another topic in the training is the Transformer model we tried to make. (The program code is in the GitHub repository). We implemented all the previously showed methods for the new solution as well, but we did not count with the large needs of memory when training this type of network. When we tried to train it like all the ResNet50 based network, we ran out of memory (we have a computer with 16GB of RAM), so we had to lower the resolution of the pictures from 256*256 to 64*64.

At this resolution, we could only implement a 16*16 patch learning, which is not big enough for the learning to be significant, Therefore we could not achieve more than 5% accuracy on the 50 class training dataset.

The experiment was really interesting of course, so it was worth the effort we put in it.

V. Hyperparameter Optimization

As the loss improvement slowed down with the default learning rate, we adopted a strategy to reduce the learning rate to 40% whenever the validation loss did not decrease compared to the previous epoch.

EarlyStopping was also implemented; if the loss did not decrease for two consecutive epochs, we halted training and restored the weights of the best-performing model. The same procedures were applied during the 10 epochs when only the final dense layers were trained and the subsequent 10 epochs when all layers became trainable.

VI. Evaluation

In the case of numerous categories, accuracy may not adequately characterize model performance. Therefore, we developed a function to calculate, for a given 0<p<1, the fraction of test data points where the predicted label falls within the top p% of all categories. This calculation was performed for various p values, and the results were visualized on a graph. This is equivalent to as the model had more guesses than 1.

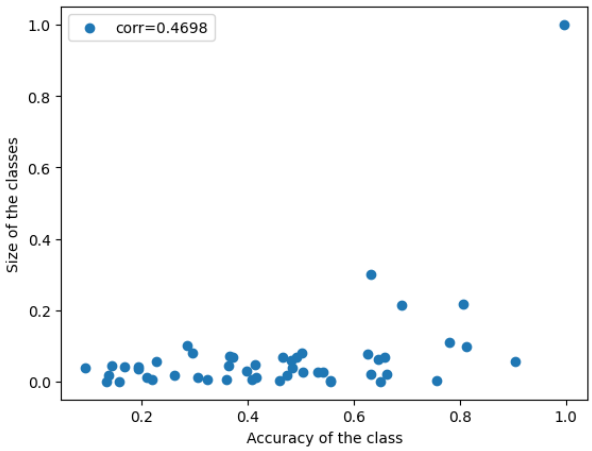

Fig. 8. Accuracy of the class based on the class-size

We examined whether there was a correlation between class size and how well the model learned them. Interestingly, although stamp images had the largest class size and performed the best, there was no significant correlation between the two quantities.

1. ResNet50 for 50 classes: Top 1 accuracy was above 0.5, indicating relatively good model performance compared to the many categories. Top 10% accuracy was over 0.75.

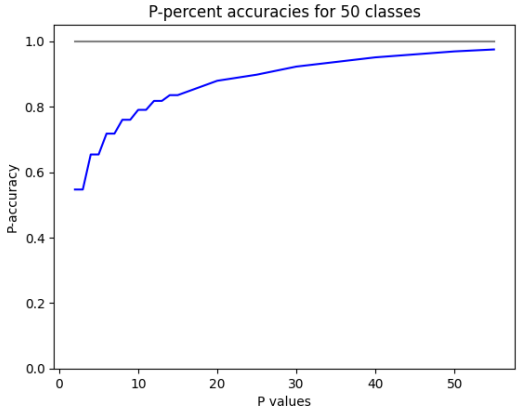

Fig. 9. P-percent accuracies for 50 classes with ResNet50

We also analyzed which classes the models recognized most and least. The model performed exceptionally well on the stamp class, with over 99% accuracy, while the lowest accuracy (0.094) was on images of dilapidated temples, which varied widely in perspective and sometimes only featured a small detail up close.

Fig. 10. Pictures of the best and worst predicted classes

2. ResNet50 for 1971 classes: Top 1 accuracy was around 0.1, and top 10% was just below 0.6. Despite including smaller classes, there was still no significant

correlation between class size and model recognition performance. The analysis of the best and worst recognized classes was based on top 1 accuracy, as using top p% led to either 0 or 1 for many classes. The stamp class remained the best recognized, but there were changes compared to the 50-category model.

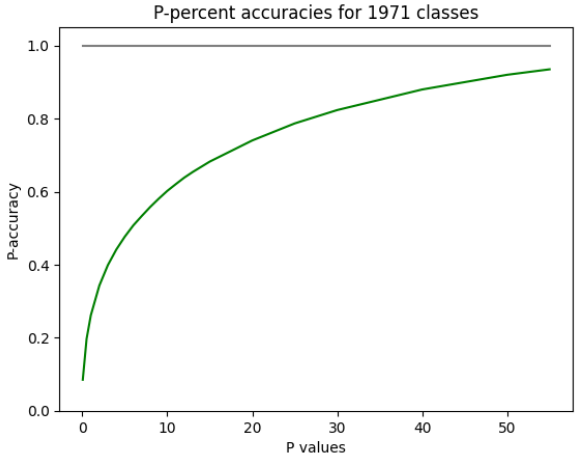

Fig. 11. P-percent accuracies for 1971 classes with ResNet50

One of the best-recognized classes included images of tanks, featuring equally sized and genuinely very similar pictures. Additionally, the least recognized class in this case is a class of depicted images of buildings, where one picture clearly shows the building, another has the same building rotated, and in some, only the people standing in front of the building are visible.

Fig. 12. Pictures of the second best and the worst predicted classes

3. Transformer for 50 classes: The transformer model faced technical limitations, requiring a significant reduction in image and batch sizes to fit into memory. As a result, achieving a meaningful accuracy was challenging. It is likely that with a machine with larger memory, the transformer model could have outperformed the ResNet50 model.

VII. Summary

In conclusion, our computational exploration in landmark recognition reflects a dynamic interplay between traditional Convolutional Neural Networks (CNNs) and the cutting-edge Transformer architectures. While challenges in implementing a full-scale Transformer-based network were encountered due to computational limitations, the endeavor provided valuable insights into the potential efficacy of such architectures. The confluence of these two paradigms has enriched our understanding of landmark recognition within the computer vision domain.

Our model training efforts, executed on a local machine with a Laptop GPU RTX3060, showcased a meticulous approach, incorporating strategies like partial training of layers and hyperparameter optimization. The evaluation metrics extended beyond conventional accuracy, providing a nuanced perspective on model performance. Findings from ResNet50 models for 50 and 1971 classes, alongside a Transformer model for 50 classes, revealed distinctive challenges and potential avenues for improvement. The nuanced evaluation approach sheds light on the intricacies of landmark recognition across diverse categories. As we anticipate future advancements with enhanced computational resources, the presented insights lay the groundwork for further refinement and innovation in the evolving landscape of computer vision.

References

[1] Ajeet Ram Pathak, Manjusha Pandey, Siddharth Rautaray: “Application of Deep Learning for Object Detection”, Procedia Computer Science, Volume 132, 2018, Pages 1706-1717.

[2] Amit Dhomne, Ranjit Kumar, Vijay Bhan: “Gender Recognition Through Face Using Deep Learning”, Procedia Computer Science, Volume 132, 2018, Pages 2-10.

[3] N. Ferracuti, C. Norscini, E. Frontoni, P. Gabellini, M. Paolanti, V. Placidi: “A business application of RTLS technology in Intelligent Retail Environment: Defining the shopper's preferred path and its segmentation”, Journal of Retailing and Consumer Services, 47:184-194, March 2019.

[4] Y. Liu et al., "A Survey of Visual Transformers," in IEEE Transactions on Neural Networks and Learning Systems, 30 March 2023

[5] Han, Kai and Xiao, An and Wu, Enhua and Guo, Jianyuan and XU, Chunjing and Wang, Yunhe, “Transformer in Transformer”, Curran Associates, Inc., Volume 34, 2021

[6] Niki Parmar, Ashish Vaswani, Jakob Uszkoreit, Lukasz Kaiser, Noam Shazeer, Alexander Ku, Dustin Tran, “Image transformer”, International conference on machine learning, pages 4055-4064, 2018

[7] Devvi Sarwinda, Radifa Hilya Paradisa, Alhadi Bustamam, Pinkie Anggia: “Deep Learning in Image Classification using Residual Network (ResNet) Variants for Detection of Colorectal Cancer”, Procedia Computer Science, Volume 179, 2021, Pages 423-431.

[8] Kaiming He, Xiangyu Zhang, Shaoqing Ren, Jian Sun: “Deep Residual Learning for Image Recognition”, Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition (CVPR), 2016, pp. 770-778.

[9] A. Victor Ikechukwu, S. Murali, R. Deepu, R.C. Shivamurthy: “ResNet-50 vs VGG-19 vs training from scratch: A comparative analysis of the segmentation and classification of Pneumonia from chest X-ray images”, Global Transitions Proceedings, Volume 2, Issue 2, 2021, Pages 375-381.

[10] Mehdhar S. A. M. Al-Gaashani, Nagwan Abdel Samee, Rana Alnashwan, Mashael Khayyat and Mohammed Saleh Ali Muthanna: “Using a Resnet50 with a Kernel Attention Mechanism for Rice Disease Diagnosis”, Life, 2023.