Multi-dimensional feature and data mining

Georgiou, T.

7 Conclusions

7.1 Conclusions

In this thesis, we explore the application of computer vision methods on high dimensional data. With high dimensional data we define data that have more than two dimensions. More specifically, we explore whether computer vision inspired approaches are capable of representing computational fluid dynamics simulation output.

In Chapter 2 we presented an extensive overview of the methodologies, data sources and applications whose subject is high dimensional vision data. The aim of this chapter is to find common practices between methodologies that, at a first glance, seem disconnected but have in common the high dimensionality of the data they process and thus answer **research question 1**: Overall we identified four common data sources, namely 2D videos, RGB-D images and videos, and 3D models such as CAD models or point clouds. Moreover, we identified common characteristics of methodologies regardless the data they are applied on and we identified two main categories of high dimensionality, high physical dimensions and large number of information per physical location. Finally, we concluded that although hand crafted feature based approaches are outperformed by deep learning based approaches, they can provide complementary information and increase the overall performance of an approach.

In Chapter 3, we constructed a large scale dataset and proposed several approaches for dealing with the high dimensionality of the data. This dataset is used as a benchmark to help us answer **research question 2**. The experiments conducted showed that deep learning approaches are well capable of representing CFD simulation output and moreover, how to better utilize our model resources, i.e., number of trainable parameters, hardware computational capabilities and available memory. Our best models are able to accurately, i.e., with about 3% accuracy, predict forces applied on

a shape not visible to them, by leveraging flow data. Meanwhile, the encoded representation is enough to reconstruct the input in a large extent leading us to believe that it still encodes most of the information of the original flow field.

In Chapter 4 we answered **research question 3**. We implemented hand crafted feature based approaches and evaluated whether they are capable of representing flow field data and how they compare to deep learning approaches. Overall we made three tests, (i) which hand-crafted feature produces the most accurate results and with what kind of sampling (i.e., dense or which detector), (ii) how does the performance compare to deep learning approaches and (iii) which approach maintains its performance better as the training set size reduces. The experimental results showed interesting behavior. The dense sampling of SIFT [225, 226] features produced the best performance out of all setups based on hand-crafted features tested. Interestingly, although in terms of $RMSE$ the SIFT approach was outperformed by the deep learning approaches, it was able to outperform them in terms of R^2. As we showed, deep learning approaches and hand crafted feature based approaches show different behavior. For most test examples, the SIFT based approach had smaller error than the deep learning one, but it also produced much larger errors than the deep learning approaches at the extreme (bringing the $RMSE$ high). This finding is strengthening the observation that hand-crafted feature based approaches can provide complementary information to the deep learning approaches.

In Chapter 5 we answered **research question 4**. By taking advantage of the vector field representation we are able to extract angle information from the input and the kernel of a convolutional layer and subsequently build rotation invariant and equivariant convolutional layers and networks. Our experiments showed that this approach produces similar classification performance to the state of the art, while having better angle prediction with a much lower computational cost.

Finally, in Chapter 6 the norm loss was proposed, answering **research question 5**. The norm loss is a soft weight regularization method for deep neural networks and it aims to keep the norm of the kernels of a convolutional layer close to one. The experimental results suggest that networks that utilize the norm Loss during training converge much faster than the equivalent that utilized weight decay. Moreover, in most cases, networks trained with the norm loss exhibit state of the art performance.

7.2 Limitations

Although the research presented here has revealed interesting information, it has also shown many limitations of the approaches proposed and tested. As the main data source for this thesis has been CFD simulation output we deem necessary to start with

this application in mind. The major limiting factor in extracting meaningful information from CFD simulation data, is its availability. CFD simulations can even produce terabytes of data. Nonetheless, most information usually comes from the same example. As a result, we have huge amounts of data, split into very few examples. This is the opposite from the ideal situation, where many examples exist and thus correlations can be found. The main reason for this limitation is the time required for CFD simulations to converge. A complicated simulation might even take a month to converge on a cluster with thousands cores. Moreover, there are many hyper parameters as well as different algorithms for obtaining CFD simulation output. With many of these parameters, such as the grid on which the flow field is computed, the result can vary significantly. Having a model capable of representing simulations produced in a different manner increases the complexity and amount of data required even further.

Regarding deep learning models, we can identify some limitations as well. A common approach to produce general high performing models is to augment the training data. By applying slight augmentations on the data during training, effectively increases the train data size by introducing new variations of the same subject, i.e., the content of the image. The resulting models become more robust to certain types and amount of noise. If a flow field is augmented using the popular approaches such as affine transformations, it will probably not be an approximation of a solution to Navier-Stokes equations anymore and more importantly it will not maintain the same label. Thus, one must be careful in how data is augmented. Moreover, many patterns appear in small scales in the flow. For the models to fit in GPU memory the input resolution has to be relatively low which may result in the disappearance of many important characteristics. Nonetheless, large scale patterns, like vortices and reverse flow still exist in the simulations examples that we utilized which made the training of models possible.

In the last two chapters, new deep learning approaches are proposed. In Chapter 5 a new operator is proposed, that utilizes the vector field representation to measure the angle between a convolutional kernel and an input signal. Utilizing this operator in deep neural networks, although shows great potential, it has several limitations. The most important one is that for most applications, the input is scalar, e.g. RGB images. Moreover, due to the $arctan$ function, during back propagation, numeric instabilities can occur which forces the usage of thresholding and masking the output. In our experience, depending on the application, the network size, the learning rates and other parameters, the most effective value of this threshold changes. This introduces another hyper parameter to the many existing in deep learning approaches. Moreover, as we saw with the current implementation, utilizing this operator on GPUs increases the computation time by a large margin, which may significantly reduce one of the main advantages of the method - lower computational requirements.

7.3 Future work

The Clifford convolution operator was implemented for two dimensional fields, as an initial case study. Most CFD simulation outputs though, have three spatial dimensions. Extending the operator to calculate solid angles might be very beneficial for minimizing the required number of free parameters in a deep learning model. Interestingly, the potential benefit of calculating solid angles might be even higher than that of the two dimensions, since the amount of convolutions needed for a max-pooling operation over solid angle increases exponentially with the number of dimensions.

This research is one of the early works on deep learning applied on CFD simulation output. One of the main limitations is the large amount of time required to produce simulations. Thus, a feasible approach in applying deep learning on more complicated simulations is transferring models. To be more precise, a model could be trained on a large scale CFD dataset that requires feasible amount of time to be created. Would this model then be applicable, with minor adjustments i.e., fine tuning, on a small dataset of complex and time consuming CFD simulations?

From a deep learning perspective, there are still many open questions. For example, one question is how to better utilize the extra dimensions of higher than two dimensional images. When a temporal dimension is included, it is not certain yet which approach is more effective. Is there a certain approach to handling the temporal dimension that suits better, or will it always depend on the application? For the static world there are similar questions that are important to research if greater performing or efficiency is required. For example, the projection to lower dimensionality, seems to be performing better than having three dimensional kernels for low computational models. Increasing the capacity of networks with three dimensional kernels, coupled with large and diverse datasets seems to perform better than the 2D projections. In many applications the computational and memory budgets are limited. It is still unclear at which point does a more complex network with high dimensional kernels is a more appropriate choice. These are questions that are becoming more relevant as computational demand of deep learning approaches seem to grow and mobile applications seem to have increasing demand.