ATLANTIS PRESS

Advances in Engineering Research, volume 146

4th International Conference on Material Engineering and Application (ICMEA 2017)

An automatic Algorithm Based on Artificial Neural Network is Applied in Taxi Target Prediction

Zhaosheng WANG[1]
Ganzhou Teachers college
Ganzhou, China
e-mail: 1239331840@qq.com

Shiyu LI[2]
Jiangxi University of Science and Techrcolog
Ganzhou, China
e-mail: 81798152@qq.com

***Abstract*—This paper describe the solution to the ECML/PKDD discovery challenge on taxi destination prediction. The work consisted in predicting the destination of a taxi based on the beginning of its trajectory, represented as a variable-length sequence of GPS points, and diverse associated meta-information, such as the departure time, the driver id and client information. Contrary to most published approaches, this paper uses an almost fully automated approach based on neural networks. The architectures we tried use multi-layer perceptions, bidirectional recurrent neural networks and models inspired from recently introduced memory networks. Our approach could easily be adapted to other applications in which the goal is to predict a fixed-length output from a variable-length sequence.**

Keywords-multi-layer perceptron; neural networks; taxi destination; prediction

I. INTRODUCTION

The taxi destination prediction challenge was organized by the 2015 ECML/PKDD conference and proposed as a Kaggle competition. The last location represents the target, and different trajectories have different GPS sequence lengths. Metadata associated with a taxi: if the client called the taxi by phone, then we have a client ID. If the client called the taxi at a taxi stand, then we have a taxi stand ID. Otherwise we have no client identification, the taxi ID, the time of the beginning of the ride. In the competition setup, the testing dataset is composed of 320 partial trajectories, which were created from five snapshots taken at different timestamps. This testing dataset is actually divided in two subsets of equal size: the public and private test sets. The public set was used through the competition to compare models while the private set was only used at the end of the competition for the final leader board. Our approach uses very little hand-engineering compared to those published by other competitors. It is almost fully automated and based on artificial neural networks.

II. APPROACH

A. *Data Distribution*

The work is to predict the destination of a taxi given a prefix of its trajectory. As the dataset is composed of full trajectories, we have to generate trajectory prefixes by cutting the trajectories in the right way. The provided training dataset is composed of more than 1.7 million complete trajectories, which gives 83480696 possible prefixes. The distribution of the training prefixes should be as close as possible as that of the provided testing dataset on which we were eventually evaluated. This test set was selected by taking five snapshots of the taxi network activity at various dates and times. This means that the probability that a trajectory appears in the test set is proportional to its length and that, for each entire testing trajectory, all its possible prefixes had an equal probability of being selected in the test set. Therefore, generating a training set with all the possible prefixes of all the complete trajectories of the original training set provides us with a training set which has the same distribution over prefixes as the test set.

B. *MLP Architecture*

A Multi-Layer Perceptron (MLP) is a neural net in which each neuron of a given layer is connected to all the neurons of the next layer, without any cycle. It takes as input fixed-size vectors and processes them through one or several hidden layers that compute higher level representations of the input. Finally the output layer returns the prediction for the corresponding inputs. In our case, the input layer receives a representation of the taxi's prefix with associated metadata and the output layer predicts the destination of the taxi. We used standard hidden layers consisting of a matrix multiplication followed by a bias and nonlinearity. The nonlinearity we chose to use is the Rectifier Linear Unit [1], which simply computes max (0; x). Compared to traditional sigmoid-shaped activation functions, the RLU limits the gradient vanishing problem as its derivative is always one when x is positive.

C. *Destination Clustering and Output Layer*

As the destination we aim to predict is composed of two scalar values, it is natural to have two output neurons. However, we found that it was difficult to train such a simple model because it does not take into account any prior information on the distribution of the data. To tackle this issue, we integrate prior knowledge of the destinations directly in the architecture of the model: instead of predicting directly the destination position, we use a predefined set (ci)i of a few thousand destination cluster centers and a hidden layer that associates a scalar value (pi)to each of these clusters. As the network must output a single destination position, for our output prediction H, we compute a weighted average of the predefined destination cluster centers:

$$H = \sum_{k=1}^{C} P_k C_k \qquad (1)$$

Note that this operation is equivalent to a simple linear output layer whose weight matrix would be initialized as our cluster centers and kept fixed during training. The hidden values (pi)i must sum to one so that H corresponds to a centroid calculation and thus we compute them using a soft max layer:

$$P_i = \frac{\exp(T_i)}{\sum_{k=1}^{C} \exp(T_k)} \qquad (2)$$

where (Tj)j are the activations of the previous layer. The clusters (ci)i were calculated with a mean-shift clustering algorithm on the destinations of all the training trajectories, returning a set of C = 3392 clusters.

D. Cost Computation and Training Algorithm

The evaluation cost of the competition is the mean Haversine distance, which is defined as follows (φ_x is the longitude of point x, λ_x is its latitude, and R is the radius of the Earth):

$$D_{haversine}(x, y) = 2R\arctan\left(\sqrt{\frac{A(x,y)}{A(x,y)-1}}\right) \qquad (3)$$

Where A(x,y) is defined as:

$$A(x, y) = sin^2(\frac{\varphi_y - \varphi_x}{2}) + cos(\varphi_x)\, cos(\varphi_y)\, sin^2(\frac{\lambda_y - \lambda_x}{2}) \qquad (4)$$

Our models did not learn very well when trained directly on the Haversine distance function and thus, we used the simpler equirectangular distance instead, which is a very good approximation at the scale of the city of Porto:

$$D_{equirectangular}(x,y) = R\sqrt{(\varphi_y - \varphi_x)^2 + \cos^2(\frac{\varphi_y - \varphi_x}{2})(\lambda_y - \lambda_x)^2} \qquad (5)$$

We used stochastic gradient descent (SGD) with momentum to minimise the mean equirectangular distance between our predictions and the actual destination points. We set a fixed learning rate of 0.01, a momentum of 0.8and a batchsize of 350.

III. Alternative Approaches

The models that we are going to present in this section did not perform as well for our specific destination task on the competition test set but we believe that they can provide interesting insights for other problems involving fixed-length outputs and variable-length inputs.

A. Recurrent Neural Networks

As stated previously, a MLP is constrained by its fixed-length input, which prevents us from fully exploiting the entire trajectory prefix. Therefore we naturally considered recurrent neural net (RNN) architectures, which can read all the GPS points one by one, updating a fixed-length internal state with the same transition matrix at each time step. The last internal state of the RNN is expected to summarize the prefix with relevant features for the specific task. Such recurrent architectures are difficult to train due in particular to the problem of vanishing and exploding gradients [3]. This problem is partially solved with long short-term memory units [4], which are crucial components in many state of the art architectures for tasks including handwriting recognition [5], speech recognition [6], image captioning or machine translation. We implemented and trained a LSTM RNN that reads the trajectory one GPS point at a time from the beginning to the end of each input prefix. We considered a variant in which the input of the RNN is not anymore a single GPS point but a window of 4 successive GPS points of the prefix. The window shifts along the prefix by one point at each RNN time step.

B. Memory Networks

Memory networks have been recently introduced as an architecture that can exploit an external database by retrieving and storing relevant information for each prediction. The encoders are the same as those of our previous architectures except that we stop at the hidden layer instead of predicting an output. This results into fixed-length representations in the same vector space so that they can be easily compared. Then we compute similarities by taking the dot products of the prefix representation with all the candidate representations. Finally we normalize these m similarity values with a soft max and use the resulting probabilities to weigh the destinations of the corresponding candidates. In other words, the final destination prediction of the prefix is the centroid of the candidate destinations weighted by the soft max probabilities.

IV. Experimental Results

A. Custom Validation Set

We obtained these new testing and validation sets by extracting random portions of the original training set. The validation dataset is used to early-stop our training algorithms for each model based on the best validation score, while the testing dataset is used to compare our different trained models.

B. Results

The testing scores of our various models on our custom testing dataset are as well as on the competition ones. The different hyper parameters of each model have been tuned. The results prove that embeddings and clusters significantly improve our models. The importance of embeddings can also be confirmed by visualizing them. 2D T-SNE projections for two of these embeddings and clear patterns can be observed, proving that quarters of hour and weeks of the year are important features for the prediction. All the models we have explored are very computationally intensive and we thus had to train them on GPUs to avoid weeks of training. Our competition winning model is the least intensive and can be trained in half a day on GPU. On the other hand, our recurrent and memory networks are much slower and we believe that we could reach even better scores by training them longer.

V. CONCLUSION

We introduced an almost fully-automated neural network approach to predict the destination of a taxi based on the beginning of its trajectory and associated metadata. Our best model uses a recurrent bidirectional neural network to encode the prefix, several embeddings to encode the metadata and destination clusters to generate the output. One potential limitation of our clustering-based output layer is that the final prediction can only fall in the convex hull of the clusters. A potential solution would be to learn the clusters as parameters of the network and initialize them either randomly or from the mean-shift clusters. Concerning the memory network, one could consider more sophisticated ways to extract candidates, such as using an hand-engineered similarity measure or even the similarity measure learnt by the memory network. In this latter case, the learnt similarity should be used to extract only a proportion of the candidates in order let a chance to candidates with poor similarities to be selected. Furthermore, instead of using the dot product to compare prefix and candidate representations, more complex functions could be used.

ACKNOWLEDGMENT

This work was supported by Jiangxi provincial education department science and technology research project (GJJ161333).

REFERENCES

[1] Alan Graves, Abdel-rahman Mohamed, and Geoffrey Hinton. Speech recognition with deep recurrent neural networks. In Acoustics, Speech and Signal Processing(ICASSP), 2013 IEEE International Conference on, pages 6645-6649. IEEE, 2013.

[2] Xavier Glorot, Antoine Bordes, and Yoshua Bengio. Deep sparse rectifier neural networks. In International Conference on Artificial Intelligence and Statistics,pages 315-323, 2011.

[3] Patrick Doetsch, Michal Kozielski, and Hermann Ney. Fast and robust training of recurrent neural networks for offline handwriting recognition. In Frontiers in Handwriting Recognition (ICFHR), 2014 14th International Conference on, pages279-284. IEEE, 2014.

[4] Dzmitry Bahdanau, Kyunghyun Cho, and Yoshua Bengio. Neural machine translation by jointly learning to align and translate. arXiv preprint arXiv:1409.0473,2014.

[5] Hasim Sak, Andrew Senior, and Françoise Beaufays. Long short-term memory based recurrent neural network architectures for large vocabulary speech recognition.arXiv preprint arXiv:1402.1128, 2014.

[6] Alex Graves, Marcus Liwicki, Santiago Fernández, Roman Bertolami, Horst Bunke,and Jürgen Schmidhuber. A novel connectionist system for unconstrained handwriting recognition. Pattern Analysis and Machine Intelligence, IEEE Transactions on, 31(5):855-868, 2009.

[7] Kelvin Xu, Jimmy Ba, Ryan Kiros, Aaron Courville, Ruslan Salakhutdinov,Richard Zemel, and Yoshua Bengio. Show, attend and tell: Neural image caption generation with visual attention. arXiv preprint arXiv:1502.03044, 2015.