Advanced Concepts of Machine Learning: Using an existing framework for a recurrent neural network

Kevin Trebing (i6192338), Alberto Perez (i6201157)

November 11, 2018

1 Set-up

We used a framework called *textgenrnn* from Max Wolf[1]. It is based on char-rnns[2] and implemented in Tensorflow. To install it Tensorflow is required (*'pip3 install tensorflow'* or *'pip3 install tensorflow-gpu'* for the GPU version). To install textgenrnn execute *'pip3 install textgenrnn'*.

2 How to use

In order to train an RNN, execute *training.py*. Here the model and training parameters are set. In our case we use a 4 layer deep RNN in which each layer has 128 neurons. As training parameters we set it to 6 epochs since our dataset is quite large and more epochs did not improve the results by a lot. Other important parameters are *max_gen_length*, which is the number of characters that will be the output and *word_level* which should be set to *False* in our case, because we want to train on character level instead of word level. The other parameters are kept at their default values. While training, three files will be created: a config file, a vocabulary file and the weights of the network. These can be used to generate text later on or resume a failed training.

In order to generate text execute *textgeneration.py*. Here we specify the config, vocabulary and weight files from our previous training and then call the function *generate()* with the keyword argument *temperature*. The temperature can be seen as a creativity parameter. Low temperature values create words where it is confident they exist, but not very interesting sentences (a lot of repetitions). Higher temperature values generate more interesting sentences, but some words do not exist. Examples can be seen in 4.

Training time of the RNN depends on the model parameters and the dataset. We used a colaboratory notebook which utilizes a Tesla K80 for the training. Training took about 45 minutes per epoch. But when *max_length* and *max_gen_length* are lowered, the time decreased to 26 minutes per epoch. Text-generation was done on our own PC.

[1] `https://github.com/minimaxir/textgenrnn`

[2] `http://karpathy.github.io/2015/05/21/rnn-effectiveness/`

3 Dataset

In order to train the network correctly we needed a txt file with the training examples. We found a dataset containing all NIPS papers on Kaggle[3]. After downloading the csv file we extracted the abstracts and removed the entries where no abstract was given. The resulting 7241 abstracts were saved per line in a txt file. Our goal was to generate abstracts that have machine learning as content.

4 Examples

Here are some examples with different temperature parameter:

Temperature 0.1:

"We present a novel algorithm for the problem of learning a set of structured prediction and statistical properties of the state of the art algorithms for the state-of-the-art methods for the state-of-the-art algorithms for the state-of-the-art algorithms."

Temperature 0.3:

"We present a novel algorithm for estimating the complexity of the original results on the context of multiple tasks. We demonstrate the effectiveness of the proposed method for solving the sample complexity and the distribution of the previous approaches that are established."

Temperature 0.4:

"This paper proposes a new model for few complexity bounds for the context of policy gradient descent on the dependence on the similarity between the data."

Temperature 0.5:

"We investigate the proposed algorithm for a single image analysis of image and the time series, where the recording to the task of inference takes in the distribution of the state-of-the-art methods in the context of tasks which can be applied to the true loss functions."

"We present a novel multi-agent problem of learning previous expected regularization in the proposed approach on the same as well as the most of the data sets and provide multiple inference algorithms for the family of message passing algorithms for the complex model for complex distributions."

Temperature 0.6:

"We present a general framework for text constraints in the data, which incorporates the latent variable model in the number of applications of the true variance of the designed subspace selection method for learning convergence rates for face in the context of a special case."

[3] `https://www.kaggle.com/benhamner/nips-papers`

Temperature 0.7:

"We propose a new approach to maximum norm may be completed to an initial system for the classic model performance on the algorithm scales algorithm. We show that our method has necessary to potentially refine the diffusion of systems in the corresponding level of classes."

Temperature 0.9:

"Neural networks, another dCOCV is a discrete or outperforming the classic model which modify a general computational complexity of timescales. We propose a new probabilistic model for probabilistic principles for images to better the states efficiency on novel uneven of algorithms with two results."

Temperature 1.3:

"Experts models are almost optimal."

Temperature 1.5:

"Quations over versions of the 'coin"',fo) 1) over a data-drives nuisance, and/orm view order-rekile errors. Motiss are revenue, ES, 2w winner–music fairness, {\em object [Ex is}] mitic fream fMRI trajectory path typically time."

4.1 Remarks

We can see that with low temperatures the generation gets stuck in a loop. Higher values create words that do not exist or are latex formats. At temperatures between 0.4 and 0.7 the most interesting sentences are generated.