The Construction of the Algorithm Study Based on the Mathematical Model of Motion

Maryna Graf[1[0000-0003-4873-548X]] and Volodymyr Kvasnikov[1[0000-0002-6525-9721]]

[1] National Aviation University, pr. Kosmonavta Komarova, 1, Kiev, 03680, Kyiv, Ukraine
graf.maryna@gmail.com, kvp@nau.edu.ua

Abstract. The purpose of this work is the construction of the algorithm study a neural network based on the mathematical model of the motion in remotely piloted aircraft systems (RPAS) or unmanned vehicle aircraft (UAV). Information technologies are considered in the UAV control system to provide two-way information transfer between the on-board computer UAV and the operator.
The problem arises in the analysis of big amounts of data with information which come from the operator to the on-board computer and in the opposite direction, as well as with a constant change under the influence of external factors. In case of distorted data transmission or collision with obstacles a hang-up and drop of the UAV is possible.
Taking into account the rapid growth of UAV usage for civilian and military purposes, the neural network training algorithm for processing the input signal is offered. It can facilitate the task of data analysis and reduce the likelihood of uncertain situations. This algorithm is designed to predict the development of the situation, increase accuracy, the rate of information transfer and its reliability.

Keywords: unmanned aerial vehicle, information technology, data transmission, algorithm study, big amounts of data.

1 Introduction

To build the neural network of the algorithm study for two-way information transfer, it is necessary to collect and prepare data for training; determine the parameters of the model for training; implement the model in the form of program code; train the model on the collected data; test the model.

The RPAS or UAV includes its own control system, which usually also contains an engine. The control system solves the problem of controlling a particular engine in order to achieve the given values of the generalized coordinates. The task for controlling the engines is to bring the BTS to the trajectory control level, where the task of calculating the generalized coordinates and selecting the trajectory for achieving the specified position or position is solved.

CEUR-WS.org/Vol-2105/10000235.pdf

1.1 Data Collection

The main stage in the construction the algorithm of study is the collection and preparation of data for learning the model. Required data for the collection, storage and processing of information between the operator and the on-board computer of the UAV - the altitude, latitude, longitude and turns in the Euclidean coordinate system relative to the UAV coordinate system, known as Euler angles: pitch, roll and yaw.

Preliminary processing of the data usually improves the results, so it is worth deleting obviously wrong combinations. After that, the information is divided into parts (height, angles of rotation, coordinate values). Next, a dictionary is created, the data is transferred to key arrays, which will be the input data for the information transfer model (dialog model).

2 Theoretical Basics

2.1 Dialog Model

In this work, the Sequence-to-Sequence dialog model, proposed by Google, is implemented, which is successfully used in the task of machine translation. Training takes place with the teacher, the input is given a sequence of tokens (the phrase in the original language), output too (translation) [1].

This model can also be used for information transfer (dialog model). The difference is that instead of translating from one language to another, one language is used. The question and answers should be the same, the axis is the time axis.

2.2 The Algorithm of Study

In this work, the algorithm of study the recurrent neural network Long short-term memory (LSTM) is implemented [2] (see Fig. 1).

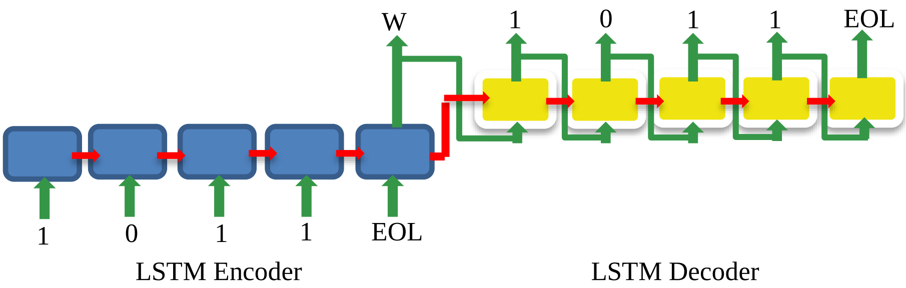

Fig. 1. Neural Network Architecture

Long Short Term Memory networks – usually just called “LSTMs” – are a special kind of Recurrent Neural Networks (RNN), capable of learning long-term dependencies. They were introduced by Hochreiter & Schmidhuber (1997), and were refined and popularized by many people in following work [3].

LSTMs are explicitly designed to avoid the long-term dependency problem. Remembering information for long periods of time is practically their default behavior, not something they struggle to learn.

Other attempts to avoid the long-term dependency problem include the use of powerful second order optimization algorithms [4][5], regularization of the RNN's weights that ensures that the long-term dependency does not vanish [6], giving up on studying the recurrent weights altogether [7][8] and a very careful initialization of RNN's parameters [9].

All recurrent neural networks have the form of a chain of repeating modules of neural network. In standard RNNs, this repeating module will have a very simple structure, such as a single tanh layer (hyperbolic tangent).

LSTMs also have this chain like structure, but the repeating module has a different structure. Instead of having a single neural network layer, there are four, interacting in a very special way.

The main feature of recurrent networks in comparison with traditional ones is the presence of feedbacks, by means of which signals, which are some sequence, from the outputs of these neurons are fed to their inputs. The values of the sequence are transmitted along with the predictions until the sequence ends.

Unlike simple recurrent networks, LSTM has three filters (input, output, forget), with which the signal is controlled inside a neuron [10] (see Fig. 2).

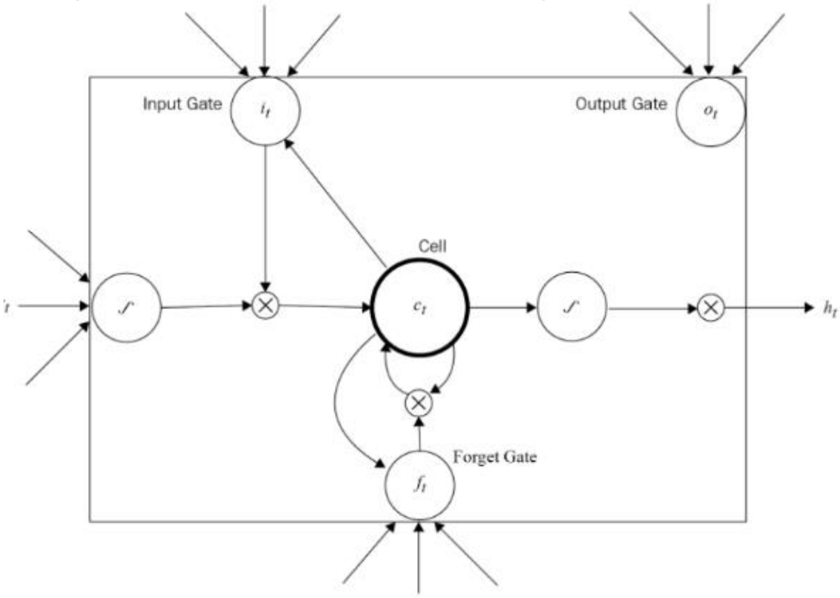

Fig. 2. Long Short Term Memory neuron

3 Using LSTM for the Construction of the Algorithm Study

The parameters of the signal include the accuracy and speed of information transfer. By processing the input signal properly, you can significantly simplify the task of analyzing information, its accuracy, rate and reliability of transmission. Manually adjusting the signal parameters when changing the characteristics of information in time, requires a lot of time. Therefore, for signal processing, its analysis depends on the influence of external factors, the signal must first be processed. One of the approaches to solving this problem is the construction of an expert model and algorithms for the logical derivation of solutions using certain methods of adjusting the signal parameters [11].

Another approach is the implementation of the neural network training algorithm to provide reliable two-way information transfer between the on-board computer of the UAV and the operator.

3.1 Mathematically

Let's describe the LSTM additions mathematically. At time t, we receive a new input x_t. We also have our long-term and working memories passed on from the previous time step, ltm_t-1 and wm_{t-1} (both n-length vectors), which we want to update.

We'll start with our long-term memory. First, we need to know which pieces of long-term memory to continue remembering and which to discard, so we want to use the new input and our working memory to learn a remember gate of n numbers between 0 and 1, each of which determines how much of a long-term memory element to keep. (1 means to keep it, 0 means to forget it entirely).

We can use a small neural network to learn this remember gate.

$$remember_t = \sigma(W_r x_t + U_r wm_{t-1}) \tag{1}$$

Next, we need to compute the information we can learn from x_t, i.e., a candidate addition to our long-term memory.

$$ltm'_t = \phi(W_l x_t + U_l wm_{t-1}), \tag{2}$$

ϕ is an activation function, commonly chosen to be *tanh*.

Before we add the candidate into our memory, though, we want to learn which parts of it are actually worth using and saving.

$$save_t = \sigma(W_s x_t + U_s wm_{t-1}) \tag{3}$$

Combine all these steps. After forgetting memories we don't think we'll ever need again and saving useful pieces of incoming information, we have our updated long-term memory.

$$ltm_t = remember_t \circ ltm_{t-1} + save_t \circ ltm'_t, \qquad (2)$$

where ∘ denotes element-wise multiplication.

Next, let's update our working memory. We want to learn how to focus our long-term memory into information that will be immediately useful. Let’s study a focus/attention vector.

$$focus_t=\sigma(W_f x_t+U_f wm_{t-1}) \tag{5}$$

Our working memory is then:

$$wm_t = focus_t \circ \phi(ltm_t) \tag{6}$$

In other words, we pay full attention to elements where the focus is 1, and ignore elements where the focus is 0.

3.2 Structure of the Software

For realization in the information storage unit, a matrix of values of the required quantities is created depending on the time. The values for storage are: height, latitude, longitude, pitch, roll and yaw, the time period is set to 1, 2 and 3 seconds.

Table 1. Matrix in the information storage unit

	height	latitude	longitude	pitch	roll	yaw
1 sec	a_{11}	a_{12}	a_{13}	a_{14}	a_{15}	a_{16}
2 sec	a_{21}	a_{22}	a_{23}	a_{24}	a_{25}	a_{26}
3 sec	a_{31}	a_{32}	a_{33}	a_{34}	a_{35}	a_{36}

The result of the operation of the information analysis algorithms and the neural network training algorithm depends directly on the input data. I.e. from the signal, which corresponds to the exact parameters.

As a neural network, there are 100 cells LSTM.

The received information is broken by element, and we learn the recurrent neural network to predict the next bit of information based on the previous bits.

Having learned a neural network, we can perform predictions of the next chain in this way.

We select the initial symbol. You can take just anyone with equal probability, or consider the probability of the appearance of symbols in the transmission / receipt. Either take the frequencies of the initial transmission symbol.

Then, in the loop, take the previously selected symbol and skip it through RNN, obtaining the output vector of the forecast.

The output layer works by softmax activation, so that the predicted neural network forecasts correspond to the probability of appearance of characters from the set.

We generate a random number [0,1) and choose a symbol according to these probabilities. Add the selected symbol to the chain and repeat the procedure a number of times.

To improve the code, each sentence on which RNN is learning is framed by two special characters <begin> and <end>.

To start generating a new sentence, it is enough to apply a token <start> to the input of the neural network. And the neural network itself knows which symbols usually come first. Also, if the choice on the next step falls on the symbol <end>, then we interrupt. This provides a more beautiful ending for the sequences being created.

For development, the Google Tensorflow library [12] was used. The function that initializes the Sequence-to-Sequence model is shown below:

```
decoderOutputs, states =
tf.contrib.legacy_seq2seq.embedding_attention_seq2seq(
      self.encoderInputs,
      self.decoderInputs,
      encoDecoCell,
      self.textData.getVocabularySize(),
      self.textData.getVocabularySize(),
      embedding_size=self.args.embeddingSize,
      output_projection=outputProjection.getWeights()
      if outputProjection else None,
      feed_previous=bool(self.args.test) )
```

The LSTM cell creation code is:

```
def create_rnn_cell():
      encoDecoCell = tf.contrib.rnn.BasicLSTMCell(
          self.args.hiddenSize,
      )
      if not self.args.test:
          encoDecoCell = tf.contrib.rnn.DropoutWrapper(
              encoDecoCell,
              input_keep_prob=1.0,
              output_keep_prob=self.args.dropout
          )
      return encoDecoCell
```

self.encoderInputs - list of the number of neurons on the hidden layer.

self.decoderInputs - a list with the same length as self.encoderInputs, but it is used as a decoder in the sequence2sequence model.

self.textData.getVocabularySize () - the number of unique tokens.

self.args.embeddingSize - the dimension of each sequence.

encoDecoCell - cell type (neuron), maybe both LSTM, and GRU.

tf.contrib.rnn.DropoutWrapper - during the network training, the random part of the neurons does not participate in the prediction, which allows the model not to retrain.

Symbols of the beginning and the end of the chain are added by hand, since they are not in the library explicitly.

chars_set.add('\r')

chars_set.add('\n')

Parameters for creating this model:

- maximum sequence length - 20 conversations;
- number of hidden layers - 6;
- the number of neurons on each hidden layer is 64;
- the number of examples that must pass through the network before updating the scale - 32;
- number of epochs – 20.

3.3 Training Model

Training recurrent neural with the help of the CPU requires a lot of resources and, as a consequence, takes a long time. Therefore, the training of the model will be conducted on the cloud service. For example, from the company Amazon [13]. With its help, the training of such a model of a long 20 it is possible to train in 10-15 seconds for an epoch.

3.4 Results

The model was tested on similar data during signal transmission.

Data for training was taken on 100 cells of the LSTM model, which is not enough, and the model constructed did not always correspond correctly, the prediction algorithm was not always accurate.

For example, in order to obtain reliable results, more than 1 million dialogues were used in problem [14].

4 Conclusions

This work describes the construction of an algorithm based on the mathematical model of UAV motion, the process of collecting data for constructing an algorithm. It describes the approach to the implementation and training of the dialogue model using the LSTM network, the basic principles of the work of recurrent neural networks. A mathematical description is given and a program is written for using LSTM to build the learning algorithm.

The structure of the long-term memory of the neural network is developed.

An algorithm for training the neural network is proposed to provide reliable two-way information transfer between the on-board computer of the UAV and the operator.

An algorithm for working with the determination of the probability of the appearance of symbols in the transmission/receipt is given.

The constructed matrix for the information storage unit. As a neural network, 100 LSTM cells are taken. The main parameters for creating such an algorithm of training based on the mathematical model of motion for UAV are described.

The Google Tensorflow library was used for development.

To improve the results of the dialogue model it is necessary to collect more data, apply word2vec technology for data preparation [15], optimize the model parameters (number of layers, number of neurons).

In the future, scientific research should be directed at increasing the speed of information processing.

References

1. Sutskever, I., Vinyals, O., Le, Q.V.: Sequence to sequence learning with neural networks. In: Advances in Neural Information Processing Systems, Location (2014).
2. Greff, K., Srivastava, R.K., Koutník, J., Steunebrink, B.R., Schmidhuber J.: LSTM: A Search Space Odyssey. IEEE Transactions on Neural Networks and Learning Systems 28, 2222-2232 (2017)
3. Hochreiter, S., Schmidhuber, J.: Long short-term memory. Neural Computation 9 (8), 1735–1780 (1997).
4. James, M.: Deep learning via hessian-free optimization. In: Proceedings of the 27th International Conference on Machine Learning (ICML-10), pp. 735–742 (2010).
5. Martens, J.,Sutskever, I.: Learning recurrent neural networks with hessian-free optimization. In: Proceedings of the 28th International Conference on Machine Learning (ICML-11), pp. 1033–1040 (2011).
6. Pascanu, R., Mikolov, T., Bengio, Y.: On the difficulty of training recurrent neural networks. In: Cornell University Library. Computer science, Learning, (2013).
7. Jaeger, H.: The echo state approach to analysing and training recurrent neural networks-with an erratum note. Bonn, Germany: German National Research Center for Information Technology GMD Technical Report, 148:34 (2001).
8. Jaeger, H., Haas, H.: Harnessing nonlinearity: Predicting chaotic systems and saving energy in wireless communication. Science 304(5667), 78–80 (2004).
9. Sutskever, I., Martens, J., Dahl, G., Hinton, G.: On the importance of initialization and momentum in deep learning. In: Proceedings of the 30th International Conference on Machine Learning (ICML13), pp. 1139–1147 (2013).
10. Graves, A.: Generating sequences with recurrent neural networks. In: Cornell University Library. Computer science, Neural and Evolutionary Computing, (2013).
11. Graf, M.S.: Metod avtomatychnogo pidboru sposobu koryguvannya tochnosti ta shvidkosti peredachi informacii v bezpilotnomu povitryanomu sudni. VII Miznarodna naukovo-tehnichna konferenciya “ITSEC”. Zbirnic tez,.pp. 46-47 (2017) (in Ukrainian).
12. Abadi, M., Agarwal, A., et al.: TensorFlow: Large-scale machine learning on Heterogeneous distributed systems. In: Cornell University Library. Computer science, Distributed, parallel, and cluster computing, (2015).
13. AWS Documentation, Regions and Availability Zones. Homepage, https://docs.aws.amazon.com/AmazonRDS/latest/UserGuide/Concepts.RegionsAndAvailabilityZones.html, last accessed 2014/10/31.
14. Auli, M., Galley, M., Quirk, C., Zweig, G.: Joint Language and Translation Modeling with Recurrent Neural Networks. In: CONFERENCE 2013, EMNLP, pp. 1044-1054. Seattle, Wasington (2013).
15. Mikolov, T., Corrado, G., Chen, K., Dean, J.: Efficient Estimation of Word Representations in Vector Space. In: Cornell University Library. Computer science, Computation and language, (2013).