MSC 68T10 **DOI: 10.14529/jcem180207**

AN INVESTIGATION OF INFORMATION INTELLIGENCE RETRIEVAL MODEL IN LOCAL SEARCHING SYSTEMS

T. Yu. Olenchikova[1], olenchikovati@susu.ru,
D. L. Maslennikov[1], et1622mdl80@susu.ru,
A. D. Marchenko[1], et1622mad76@susu.ru.
[1] South Ural State University, Chelyabinsk, Russian Federation.

The process of building and learning neural networks requires considerable computational effort. Fortunately, nowadays there are many various systems with high computing power. Development of systems, algorithms, methods and their applications in machine learning has become a promising direction in science. The work is devoted to the investigation of information retrieval problems and the main problems associated with this task, constructing an overfit-safe neural network model for improving the local searching system quality on 49% using machine learning algorithms.

Keywords: information retrieval; searching system; machine learning; neural networks; mathematical modeling; relevance increasing.

Introduction

Every day the amount of information in the world increases. It is impossible to process large datasets manually, so we can see an increasing amount of data processing algorithms and new problems, one of which is information retrieval [1]. Often, all available data cannot be structured, and we have to process heterogeneous data. It is impossible to store heterogeneous data in traditional databases, so information retrieval problems appear. One of the main problems of information retrieval is the choice of the relevance criteria. These criteria are defined by certain function of lots of parameters such as frequency of occurrence of a word, location of word in document, distance between words and other. Some relevance function may approximate result well in a certain situation, but in another situation results may be erroneous. For example, one is searching information in a set of scientific articles. The relevance function of word location in document can work well since keywords, header, abstract are located at the beginning of document. But in the situation with tag searching the order of words in document is not substantial. So this function can give bad results. Complex searching systems can use comprehensive algorithms and models connected with different branches of science such as natural language processing, topic modeling and other. Any searching system is a solution of information retrieval problem, but another challenge is to improve the output results of this system. Intelligence retrieval implies the using of machine learning for increasing quality of searching system. We use a neural network [2] based model. Neural network in substance is one of the ways to organize associative memory. Network processes input set of parameters into some other signal. Type of processing inside the network depends on architecture of the network and strength of connections between elements of the network. Strength of connections is defined by special optimization algorithms. The process of calculating these connections is called fitting (training). Neural networks are used in many areas of science when there arise problems of classification, clusterization, prediction and other. Our model is based on

T. Segaran's approach [3]. His approach has some drawbacks such as ineffective method of learning, ineffective type of training data building and overfitting predisposition. We try to construct the model without these drawbacks.

Problem stating: There is a data storage. Each object has a text description and a key. The description is a list of tags and we call it a document. The key is a path to document in the storage. There is a searching system for this storage L(Q), where Q is searching query. The output result of the system is a list of documents and their keys in order of system's relevance. The goal is to improve the relevance function using the information from users of system. Note that the system is collecting all information about query, results and selections. We need to solve the following problems:

1. Construct the model for increasing relevance.
2. Preparation of training data.
3. Model fitting.

1. Construction of the Model

In construction of searching system indexing of initial data is neaded. The easiest way to index data is to count frequency of each word in each document.

Let Q be an input query that contains words $w_1, w_2, w_3 \in W$, where W is a dictionary, received at indexing of documents. Then an input query can be defined as a vector $I = \{i_1, i_2, , i_m\}$, where m is a number of words in dictionary, $|W| = m$. This vector consists of units and zeros, moreover $i_k = 1$ when the k-th word of dictionary is in query. The same way is used to define the output data of the system. Let $O = \{o_1, o_2, , o_n\}$, where n is the number of indexed documents, $|D| = n$, D is a set of documents.

In new terms we can define problem of increasing the relevance. Let $\hat{I}$ be a set of query vectors, $\hat{O}$ be a set of documents, y^* be an objective function $y^* : \hat{I} \to \hat{O}$. The values of y^* are known only on finite number of pairs $\langle I_i, O_i \rangle (I_i \in \hat{I}, O_i \in \hat{O})$. The set of these pairs forms the training data. Notation $y^*(I_k) = O_k$ means that the input vector I_k is related to a document vector O_k. Build a model with generalizing ability. It means that the model must approximate objective function not only on training data but on a whole set of $I(\hat{I} \subset I)$. Build the model as a neural network. The i-th value of the output vector will be interpreted as a probability of relation to the i-th document of D, then we can define the loss function as a categorical cross entropy

$$Hy'(y) = -\sum_i y_i \ln(y_i) \tag{1}$$

and the error of neural network is equal to

$$-\frac{1}{|S|} \sum_{m \in S} \sum_{j=1}^{n} O_j^{*(m)} \ln(O_j^{(m)}), \tag{2}$$

where o^* is a required output, o is an actual network output, S is training data set, $|S|$ is amount of training samples. Minimization of this function (neural network fitting) will be held using the method of stochastic optimization ADAM [4].

Now define architecture of out neural network. It will be three layer forward propagation neural network. The input layer has $|W| + 1$ parameters (we add one

parameter for calculating bias, it is very handy because all actions are calculated in a matrix form). The output layer has $|D|$ parameters. The hidden layer has $\sqrt{(|W| * |D|)}+1$ parameters.

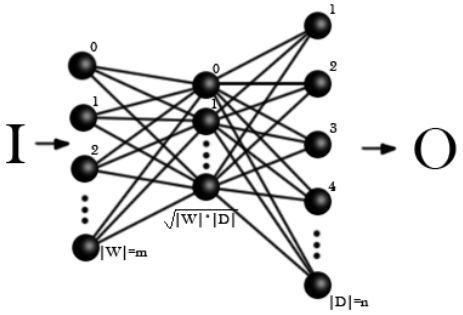

Fig. 1. Neural network architecture

For the input and hidden layers we use activation function $ReLU(x) = max(0, x)$, because it disregards problem of vanishing gradient and is simple for calculation. The output activation function is a softmax function, that transforms the signal to a vector which sum of components is equal to 1. So at the output we get a vector describing probabilities query of relations to certain documents. For disregarding problem of overfitting we use $L2$ *regularization* [5] and $Dropout(p)$ [6]. Regularization defines error penalty, and $Dropout(p)$ uses parameter p and "switches off" the neuron of current-trained layer with probability equal to p. Using dropout leads to more careful error processing on other neurons.

2. Training Sample

The method of construction of training sample described in [3] is as follows. A specific coded query generates an output vector with only one document. It means that the user chooses a specific document. For our model we will make not only one document for a specific query. We suppose that the user chooses several documents in one search session. The selection of these documents has equal probability. So the distribution of these documents selection out of the statistics will tend to normal distribution by the central limit theorem. Ideally the training set is data, formed by experts using searching system $L(Q)$. In our case we will model the users actions in searching system. Searching system gives ranking of documents. Suppose that the expert defines some documents to be relevant, but these documents are not top-ranked by the system. Name those documents as "exceptions". There are cases when the expert agrees with system and chooses top-ranked documents. These documents are named "regular". We need also some noise data and define these documents (erroneous clicks) as "ejection". We can define ratios for these types of documents. The ratio of "exceptions" and "regular" defines conditional quality of searching system (number of cases, when the expert disagrees and agrees with the system). Let ratio of "ejections", "exceptions" and "regulars" be as 9:20:1. We generate a query of random words and random length using normal distribution. Probability density graph shows distribution of words length (with expectation equal to 3 and variance equal to 1).

Generate an input query Q and get a list of documents $L(Q)$. Now for Q generate 30 outputs using scheme defined above. For the "regular" cases we generate output with

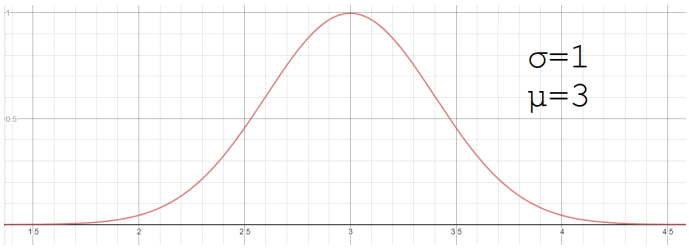

Fig. 2. Generating of words length

several top documents (most relevant by searching system). For "exception" take random documents from non-top results, and in a random way take one document for "ejection". Thereby we can model behavior of experts and generate a lot of training data.

3. Estimation Quality of the Model

Having a dictionary with 3000 words and the document set cardinality equal to 30000, amount of 10^7 training samples was generated. Split training data into two parts 80% and 20%. The first part (80%) was used for fitting the model. From the second part we take only "exception" samples and use them for testing. The error of model using (2) equals to 51%, the accuracy is 49%. We can interpret this as each 2nd case when expert disagrees with the system is taken into account. We also try to generate samples according to approach in [3]. The accuracy in this case is equal to 32% and the time of fitting is more than 1.5 times larger because we generate more training samples. Note that we get 19% of improving compared to approach in [3]. For clarity we can take subset of 10 words (and 68 related documents). Generate a training set for light version of our model. Consider the output vector of light model on histogram. The horizontal axis correspondes to the indexes of documents, vertical axis shows probability of relation to document:

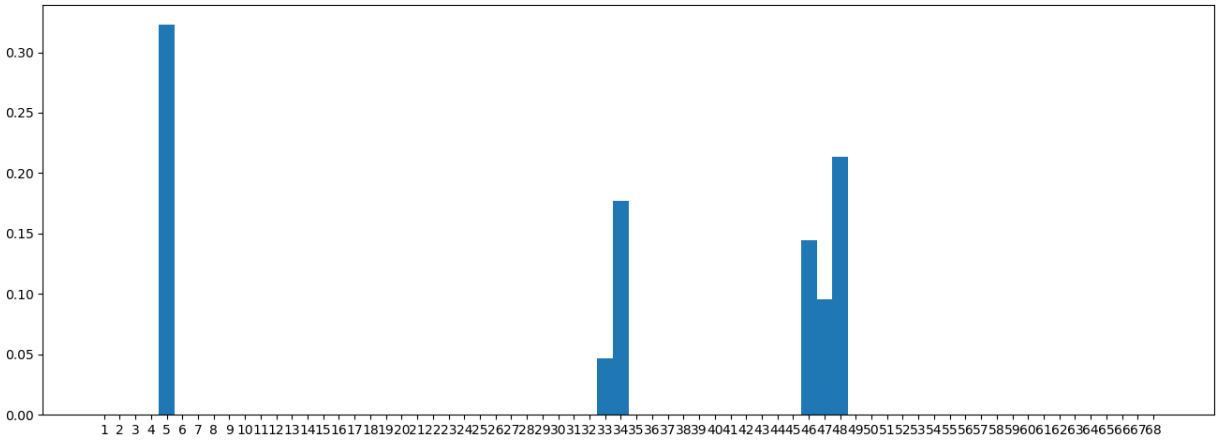

Fig. 3. Light neural network output

The output of this model is interpret as a recommendation system. For a specific query we get a probability distribution of relation to documents. Most of documents have zero values, but we also have non-zero values which mean than these documents were used by expert for this specific query. In terms of initial problem the trained neural network

predicts probabilities of using documents for specific input query. This model can be used for increasing the relevance of searching system by training on statistics, gathered from experts and improving the quality of current system. The drawbacks of the model are:

— the model works only for local searching systems with limited size of dictionary. In case of large dictionary (the larger dictionary means the larger document set) fitting time will be huge and neural network will not work in real-time;

— new gathered data is needed for re-training the model;

— insensitivity to the words order, in case of searching with natural language (not with tags as in our case) is substantial.

Conclusion

The model that increases relevance of local searching system to 49% using forward propagation neural network was constructed. We change the type of building dataset, activation functions, use regularizer and dropout to guarantee non-overfitting and improve similar models.

References

1. Manning C.D., Raghavan P., Schütze H. *Introduction to Information Retrieval*. Cambridge, Cambridge University Press, 2008. doi: 10.1017/S1351324909005129.
2. Rashid T. *Make Your Own Neural Network*. CreateSpace Independent Publishing Platform, 2016.
3. Segaran T. *Programming Collective Intelligence: Building Smart Web 2.0 Applications*. O'Reilly Media, 2007.
4. *Kingma D.P., Ba J.L. Adam: A Method for Stochastic Optimization*, available at: www.arxiv.org/pdf/1412.6980v9.pdf.
5. Krogh A., Hertz J.A. A Simple Weight Decay Can Improve Generalization. *Proceedings of the 4th International Conference on Neural Information Processing Systems*. San Francisco, Morgan Kaufmann, 1991, pp. 950–957.
6. Srivastava N., Hinton G., Krizhevsky A.,Sutskever I., Salakhutdinov R. Dropout: A Simple Way to Prevent Neural Networks from Overfitting. *The Journal of Machine Learning Research*, 2015, vol. 15, issue 1, pp. 1929–1958.

Tatyana Yu. Olenchikova, PhD (Techn), Department of Applied Mathematics and Programming, South Ural State University (Chelyabinsk, Russian Federation), olenchikovati@susu.ru.

Dmitrii L. Maslennikov, Student, Department of Applied Mathematics and Programming, South Ural State University (Chelyabinsk, Russian Federation), et1622mdl80@susu.ru.

Anton D. Marchenko, Student, Department of Applied Mathematics and Programming, South Ural State University (Chelyabinsk, Russian Federation), et1622mad76@susu.ru.

Received May 7, 2018

УДК 004.93 DOI: 10.14529/jcem180207

ИССЛЕДОВАНИЕ МОДЕЛИ ИНТЕЛЛЕКТУАЛЬНОГО ПОИСКА В ЛОКАЛЬНЫХ ПОИСКОВЫХ СИСТЕМАХ

Т. Ю. Оленчикова, Д. Л. Масленников, А. Д. Марченко

Процесс построения и обучения нейронных сетей требует немалых вычислительных трудов. К счастью, в настоящее время существует обилие различных вычислительных систем, обладающих достаточно высокой вычислительной мощностью. Разработка систем, алгоритмов и методов, их применение с использованием машинного обучения, стала перспективным направлением в науке. Работа посвящена исследованию задачи информационного поиска и основных проблем, связанных с этой задачей, разработка защищенной от переобучения нейросетевой модели для повышения релевантности поисковой системы на 49%, используя алгоритмы машинного обучения.

Ключевые слова: информационный поиск; поисковая система; машинное обучение; нейронные сети; математическое моделирование; повышение релевантности.

Литература

1. Manning, C.D. Introduction to Information Retrieval / C.D. Manning, P. Raghavan, H. Schütze. – Cambridge: Cambridge University Press, 2008.
2. Rashid, T. Make Your Own Neural Network / T. Rashid. – CreateSpace Independent Publishing Platform, 2016.
3. Segaran, T. Programming Collective Intelligence Building Smart Web 2.0 Applications / T. Segaran. – O'Reilly Media, 2008.
4. Kingma, D.J. Adam: A Method for Stochastic Optimization / D.J Kingma, J.L. Ba [Электронный ресурс]. – url: www.arxiv.org/pdf/1412.6980v9.pdf.
5. Krogh, A. A Simple Weight Decay Can Improve Generalization / A. Krogh, J.A. Hertz // Proceedings of the 4th International Conference on Neural Information Processing Systems. – 1991. – P. 950–957.
6. Srivastava, N. Dropout: A Simple Way to Prevent Neural Networks from Overfitting / N. Srivastava, G. Hinton, A. Krizhevsky, I. Sutskever, R. Salakhutdinov // The Journal of Machine Learning Research. – 2015. – V. 15, issue 1. – P. 1929–1958.

Оленчикова Татьяна Юрьевна, кандидат технических наук, кафедра прикладной математики и программирования, Южно-Уральский государственный университет (г. Челябинск, Российская Федерация), olenchikovati@susu.ru.

Масленников Дмитрий Леонидович, студент, кафедра прикладной математики и программирования, Южно-Уральский государственный университет (г. Челябинск, Российская Федерация), et1622mdl80@susu.ru.

Марченко Антон Дмитриевич, студент, кафедра прикладной математики и программирования, Южно-Уральский государственный университет (г. Челябинск, Российская Федерация), et1622mad76@susu.ru.

Поступила в редакцию 7 мая 2018 г.