Conference - Business Informatics Research

Deep Learning for Information Extraction in Finance Documents – Corporate Loan Operations

Abstract

This paper describes a knowledge -based artificial intelligence (AI) system called LODES that extracts the key data points from the loan applicants, Bill of Landing, Transport documents etc. LODES is an acronym for a decision support system (DSS) for Loan Operations Data Extraction Systems. A unique feature of LODES is that it has the capacity to learn; this is achieved by equipping LODES with an inductive inference engine that complements its deductive problem solver. The paper explains the AI system that uses inference rules to simulate the thought process of a lending maker officer when evaluating a loan request. The inductive learning approach and the learning logic of LODES are described and, additionally, there is an illustration of the system's operation in the loan evaluation process. The paper concludes with an empirical study of a LODES application.

# Introduction

Corporate lending is a complex business process. Regulations, global connectivity, heterogeneous participants and disparate operational environment of products and services delivery aggravate the complexity of this business process. Front office operations include sales pipeline, credit pipeline, regulations management and relationship management. The back office operations include loan document processing, anti money laundering checks, loan approvals and funding. In this project, our focus is to improve the back office operations.

Back office operations high level business process I depicted in the figure 1. Document processing is the stage where the business users receive a set of documents from the corporate lender. The key data points from the documents is extracted and sent to the next stage. Anti money laundering process involves laws, regulations and procedures to  prevent criminal behaviours in financial transactions. Once the checks are successful, the funding team processes the loan and funds the lender.

The key challenge in the document process is the voluminous and variety of document set. Each loan request involves more than 100 pages on an average and 10 customers on an average per day. Some customers may submit 500 page documents as well. The variety challenge includes the type of the documents; invoices, bill of lading, courier services, air way and additional supporting documents. The task of the document processing is to extract data points from such humongous datasets.

Currently, data processing is done manually by makers and checkers. A checklist of bank rules, conventional methods and personal judgment are used to extract the data points from documents. The first challenges is that this painstaking and tedious process results in loan process delay and introduces risk of human errors. The second challenge is the accuracy of the data points. For example, the customers submit documents with spelling errors, non-standard currency formats and date formats. Despite the increase in inefficiency in the process most of the banks are reluctant to use web applications to simplify document processing stage. For example, the web application is a platform for the corporate loan applications and when customers submit the applications, the data points can be easily extracted automatically with simple rules. However, the customers are reluctant to use the web applications as the customers are reluctant to move away from the traditional methods. For example, they prefer to generate the invoices from their organization systems and submit as pdf files as it is convenient for them rather than changing their systems or manually entering the data in the bank web applications.

To overcome these issues, in this paper we propose deep learning and NLP based system for document processing in loan operations. Deep learning have been fruitfully used in a variety of business fields including marketing [Chui, 2018], accounting{T sun}, healthcare [Leung] and finance (ding, Yue, hseih, Chen). Most of the studies have used neural networks for predicting stock market predictions[ding X, Yue, hsieh], ,Trading and investment applications[Chen] and credit and loan applications [Zhu]. Natural language processing (NLP) provides algorithms and techniques to analyse human language computationally. It has been used successfully for spell checks [Neha], text extraction [] and data formats [kibble].

This study aims to develop a loan operations data extraction system (LODES) using deep learning and NLP techniques. The purpose of using the deep learning algorithm in document processing stage is to simplify a makers’ and checkers’ job, to reduce human errors and to achieve more efficiency and productivity. Secondly, the study uses natural language processing techniques to address the challenge of spelling errors and non-standard formats. This study explores the power of using deep learning and NLP models in banks and the key general objectives of this study are:

1. To apply deep learning techniques to improve the efficiency in data point extraction from the voluminous document sets
2. To apply the natural language processing techniques to improve the accuracy in data point extraction from the voluminous document sets.

1. What is corporate loan operations

Business loans, are loans given to companies and other sunch entities to meet their day-to-day expenses, fund their working capital requirements and expansion etc. They are also called as corporate loans. A couple of examples could include infrastructure finance, working capital finance, term loans, letter of credit etc.

2. What is the challenge

Ensure timely and prompt processing as per committed SLA. Ensure processes and controls adheres to guidelines (internal, regulatory, etc.). Ensure timely and effective escalation of potential issues and resolution. Work closely with various internal and external stakeholders. Process end-to-end loan transactions, including and not limited to loan disbursement, documentation handling (checks, preparation and safekeeping), static data updates, repayment, refund, follow-up with internal and external stakeholders, etc. (Front Office, Compliance, Credit, etc.) to complete required processing and resolve issues arising

3. Current approaches to challenge

Most of the Financial Institutions are receiving the pdf scanned documents from the customer via email or fax. Most of the customers are reluctant to use the web based applications to send their details. The FI corporate loan operations team are manually reading the documents and entering the required data points manually into the system. It is a time consuming process.

4. How deep learning is useful for business?

As per Pareto principle (also known as the **80**/**20 rule**, the law of the vital few, or the principle of factor sparsity)( Bunkley, Nick; Box, George E.P.; Meyer, R. Daniel (1986)) states that, for many events, roughly **80**% of the effects come from **20**% of the causes. It is an axiom of business management that "80% of sales come from 20% of clients"( Marshall, Perry (2013-10-09)). If the top customers documents trained using deep learning then those volume of documents data could be extracted using deep learning. This will save considerable amount of time and the load processing will be completed within the SLA. Moreover the Loan Operations team members could focus on more value addition work.

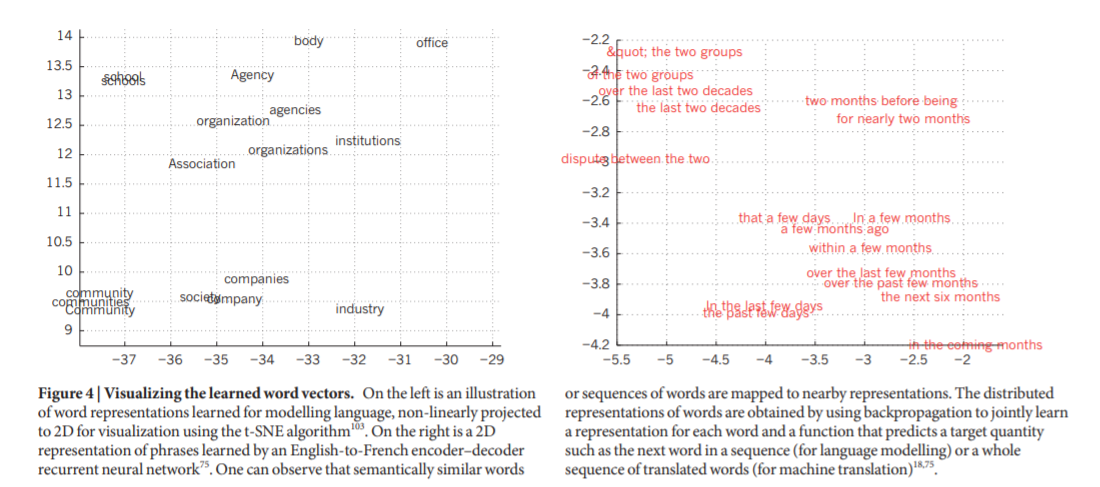
5. In this paper, we present the data point extract for loan documents.

# Background:

## Loan operations background (Business process)

## Deep Learning (Technical)

Conventional machine-learning techniques are limited in their ability to process natural data in their raw form. Constructing a pattern-recognition or machine-learning system required careful feature engineering to extract the information from documents. Representation learning is a set of techniques that allows a machine to be fed with raw data and to automatically discover the representations needed for information detection and extraction. Deep-learning methods are representation-learning methods with multiple levels of representation-learning methods with multiple layers or levels of representations or features. The key aspect of deep learning is that these layers of features are not designed by human engineers: they are learned from data using a general-purpose learning procedure. Figure 2 shows the vector representation of the learned models in the n-dimensional space.



BERT is a pre-trained deep learning natural language framework that has given state-of-the-art results on a wide variety of natural language processing tasks. This is a generic fleixible framework where the tool can be trained on new datasets. In our case, we use the labelled human datasets to further train bert framework for data point extraction.

There are two steps in BERT framework: pre-training and fine-tuning. During pre-training, the model is trained on unlabeled data over different pre-training tasks. For fine tuning, the BERT model is first initialized with the pre-trained parameters, and all of the parameters are fine-tuned using labeled data from the downstream tasks.

Iterations….

Any other details…

## Natural language processing (Technical)

Spellcheck

Document tokenization

Sliding window algorithm

The training phase is limited with the number of maiximum words or tokens. Loan application documents usally consists of more than 100 pages on average. Theerfore to train the mode, sliding window approach is used. As illustrated in the figure below, a sliding window is used to scan through the pages in the document and create a n-vector (only n words in the whole vocabulary) for each page and overlap of m words.

# Problem Definition:

## Problem Statement

Input – The input to our task is the pdf file which consists of multiple pages with information related to invoices, la… The Figure X shows the sample pdf file

Output – The output of the task is the extraction of values of specific list of data points form the document. Some example data points include; buyer name, seller name, invoice number, invoice date, currency etc. Figure Y shows the sample data points and the values.

## Challenges

What challenges are expected?

1. Various types of documents.

The documents from which the text needs to be extracted can vary from as bill of lading (BL), invoice, custom’s clearance etc.

1. The data is embedded in the text.

eg:

1. Data format varies from customer to customer

Each customer’s BLs or invoices can have different formats and it’s not feasible to have different models for each customers and each BL.

1. Date Formats challenges

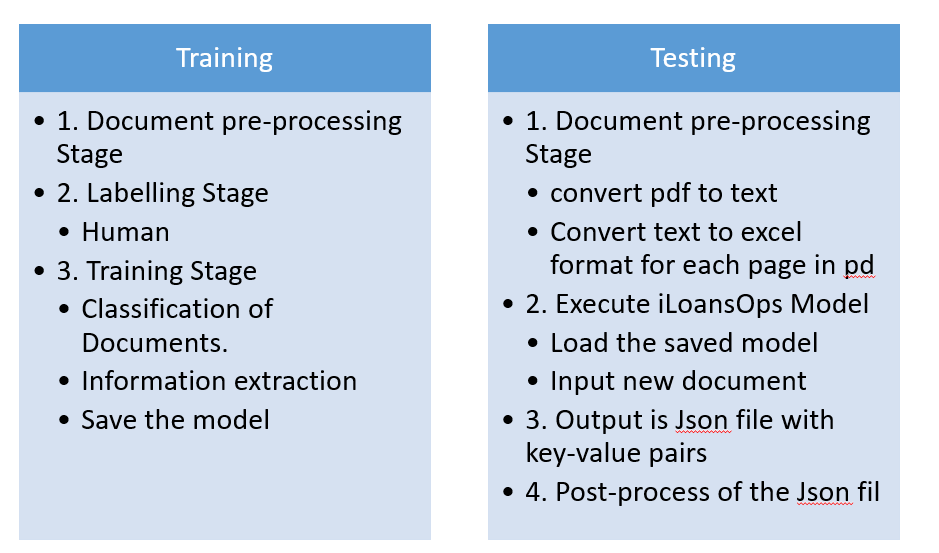
The date formats vary from customer to customer and can also vary among the different document types such as BLs, invoices, etc.

1. currency format

There can be several typos in the currency formats extracted by the OCR.

# LODES Design:

Figure 4 shows the solution design for LODES system. There are two basic phases of mode; NN learning, and testing. The system is trained on historical examples of input and output variables. During this learning process, the system learns to recognize patterns by constructing the relationship between inputs and outputs coming up with the final output. Then, a comparison between the actual and desired output occurs and errors are calculated



## Training Stage

1. Document pre-processing Stage

PDF files can be a scanned pdf or a normal pdf. In our project we use Abbyy fine reader to extract the text from the pdf document and save it in a .txt file. The extracted text file is uploaded to the system, and the system generates excel files for each page. The excel files are then used for labelling.

1. Labelling Stage

To get the labels for the examples, we used the traditional excel based approach. The label, required text and the desired output were manually labelled by the user. Figure **j1** shows how the labelling is done using excel file. The labels are categorical values with string data type, the required text is the key word or the information required for training, and the desired output is the text without any OCR extraction typos. Each file name is padded with the document type information. The labelled examples are passed to the system for training.

The data from the excel files are converted to Spacy’s annotation scheme. Algo 1 shows the pseudo code of the conversion from excel files to Spacy’s annotation scheme. Figure j2 shows the Spacy converted annotation scheme. From spacy’s annotation format we convert it to IOB tagging scheme, the pseudo code is shown in algo 2, and figure j3 shows the converted IOB format.

Later the document text is converted is converted to IOB tagging format to make it work with the text classification model.

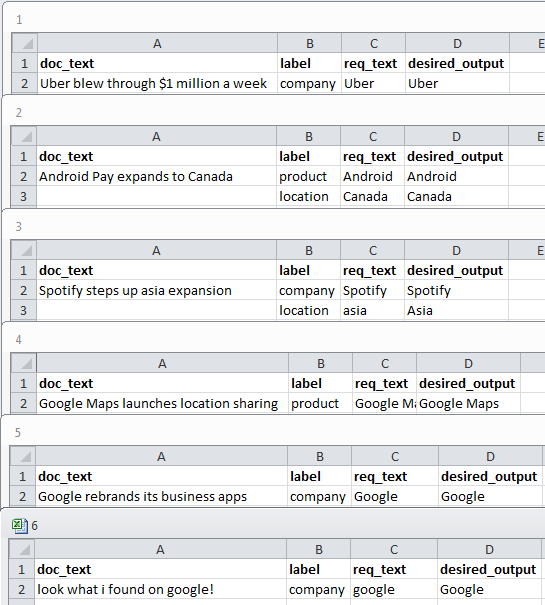
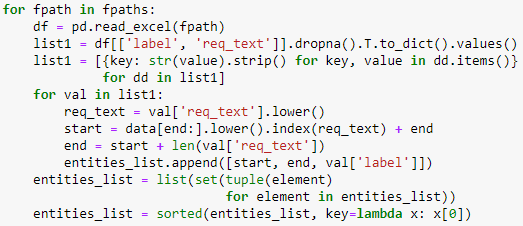


Fig j1. Users labelled excel files.

The excel files are loaded and iterated one by one. The required text is taken from excel and each required text is searched in the document to identify its index. After identifying the index of the first required text, the starting of the search space is then shifted to the end index of the first required text and so on. Finally the identified indexes are sorted based on starting index.



Code 1. Excel to spacy format.

|  |
| --- |
| **for** each filePath in directory **do**  read file and get the labels and req\_texts  **for** each label, req\_text in labels, req\_texts **do**  get index and length of req\_text from document  append the index, index + length, label to a new array  **end for**  **end for** |

Algo 1 – Excel file to Spacy’s annotation format

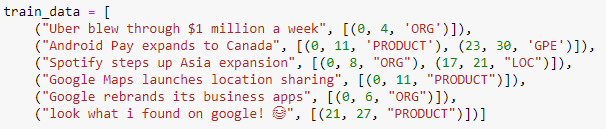
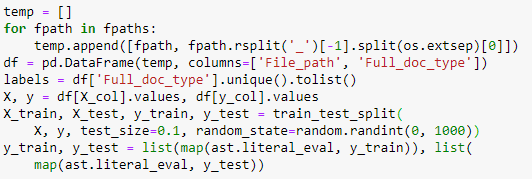


Fig j2. Spacy’s annotation format.

The excel files are loaded and iterated one by one, the file name of each excel is split using the split parameter to get the document type. The document type is then added the dataframe along with the filepath and the file contents. It is later split into train and test sets.



Code 2. Document file name to document classification data.

|  |
| --- |
| **for** each filePath in directory **do**  read filePath and get the document type  split the data for training and testing  **end for** |

Algo 2 – Preparing data for document classification

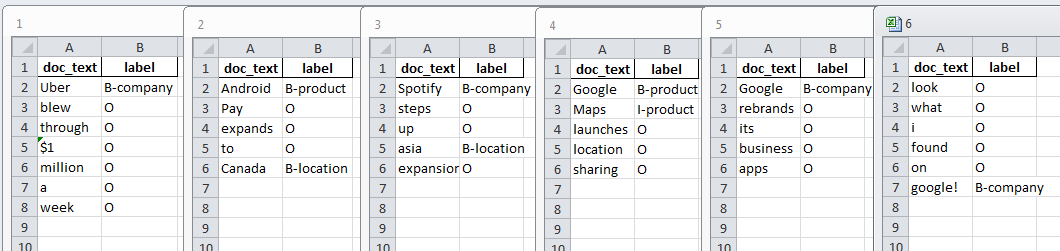


Fig j3. IOB scheme.

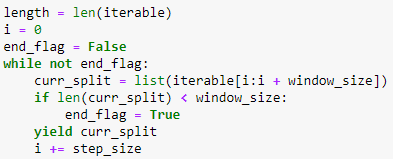
For training the document classification model initially a blank model is created. Then the training data is split into batches. Since our system uses GPU the training data is split into batches. The batches are run in parallel, for each epoch during the training process. The precision, recall, and the f-score are calculated for each epoch.



Code 2. Document classification model.

|  |
| --- |
| create blank model  create a pipe for ‘ner’ model  begin training  divide training data into batches  **for** each batch in batches **do**  get the text and labels  train the CNN model  **end for** |

Algo 3 – Document classification model

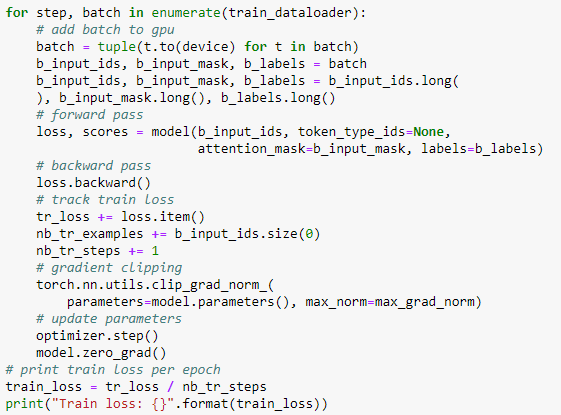


Code 3. Sliding window

|  |
| --- |
| initialize i to 0  **while** not at the end of the array **do**  curr\_slice 🡨 slice the array based on window size  **if** the curr\_slice < window\_size **then**  end iteration  **end if**  increment i  return curr\_slice  **end while** |

Algo 4 – sliding window algorithm

For the text extraction model the training data is split into batches. For each epoch the forward pass, backward pass, and gradient clipping is carried out. The f-score, validation loss, training loss, and accuracy are calculated for each epoch.



Code 2. NER model training.

|  |
| --- |
| load training data  tokenize training data into array  split the large array into smaller array using sliding window algorithm  post pad the array of tokenized texts  initialize the BERT model  split data to train and validation  **while** not at the end of the epoch **do**  train the BERT model  evaluate the model with the validation set to find out precision, recall, f-score and accuracy  **end while** |

Algo 5 – NER model training

The system groups the documents according to the document type, and is passed to the document classification model.

3. Training Stage

Information extraction [Deep learning APIs] Addresses C1,C2

For extracting the required information from the document, Bidirectional Encoder Representations from Transformers (BERT) model is used. The BERT model is used for text classification from documents. The model training is carried out in two stages, the first stage is pre-training where the model is trained using the pre-training parameters. The second stage is fine-tuning where the labels from the examples are used. The undivided architecture of BERT makes it feasible to accommodate different tasks / purposes. Encoder-decoder architecture is commonly used in several competing neural sequence transduction models [17, 18, 19]. The encoder and decoder are comprised of several layers, and each layer is responsible for mapping the symbolic input sequence to a continuous sequence representation, and vice versa.

An autoregressive model [20] is used in each layer to expend the previous value linearly, and use it as an additional input in the next stage. Each layer in encoder has two sub layers and decoder had one sub layer. In encoder, one sub layer is responsible for performing multiple attention mechanisms, and the another sub layer is a simple feed forward fully connected network. The sub layer in the decoder is used to perform multiple attention mechanism over the output from the encoder layers.

ReLu activation function is used in between the layers of the encoder and decoder. ReLu activation function runs inside the two linear transformations between the encoder and decoder layers. The attention method is responsible for mapping a group of key-value vectors and a query vector to an output vector. Multi-head attention is used in the BERT architecture as it allows every position in decoder to apply over all positions in the input sequence.

BERT architecture can accommodate tokens not more than 512; whereas our example documents were of length greater than 1000 tokens. A sliding window approach with window size 500 and step size 256 was used to fit our data into the BERT architecture. A 1000 token document was split into 4 chunks and each chunk was passed to the BERT model for training. During prediction, the chunks were joined together using a reverse sliding window approach. Probability scores of the predicted values were used to filter the overlapping values in the chunks.

1. Save the model [Reg\_DL\_Model, Unstructured DB]

The document classification model and the text extraction model are saved in the mongoDB. The models are saved along with the labelled excel files. MongoDB’s document-oriented structure makes it feasible to use.

## Testing Stage

1. Document pre-processing Stage

convert pdf to text [OCR]

Convert text to excel format for each page in pdf[Python]

2. Execute iLoansOps Model

Load the respective saved model with new document as an input

Json output with key-value pairs

3. Post-process of the Json file

Standardise Date formats [Regular expressions in Python] Adresses C4

Standartise currency formats [CurrencyConverter in Python] Adresses C5

## Visuals of Bert Training

<https://arxiv.org/pdf/1906.04341.pdf>

# Experiments

## 1. Data

The banking data set had 140 personal loan applications. Of these, 94 cases were used in the training and 46 were used in the testing. Both training and testing data sets contained half-good applications and half-bad application

Defining the network parameters: Deciding on the optimal parameters for the MLFN model is a critical issue. The best model is one that has the combination of parameters that minimizes the mean squared error.

Number of hidden neurons: The objective of the first successive trials was to hit upon the optimal number of hidden neurons that would enable the network to evaluate loan applications with the best performance.

Learning rate: This research carried out different trials to decide on a learning rate that successfully directed the degree of weight modification during the training epochs. For each trial, there were different runs of the network. The error threshold was set at 0.05 levels and the number of epochs was set at 10000. Then, the average epochs needed to reach 0.05 were calculated.

The training would stop whenever the performance of the network reached below 0.05 or when the training epochs reached 10000 even if the goal was not met.

## 2. Classification evaluations

Accuracy, F1scores

Since our model optimization required a balance on precision and recall we used F1 score to measure our accuracy. F1 score is not vulnerable to outliers, and is calculated across the distribution of all examples. The precision and recall scores are also considered for validation. All the scores are calculated for each iteration / epoch.

What failed? Analysis where the model will not perform correctly

## 3. Information Extraction evaluation

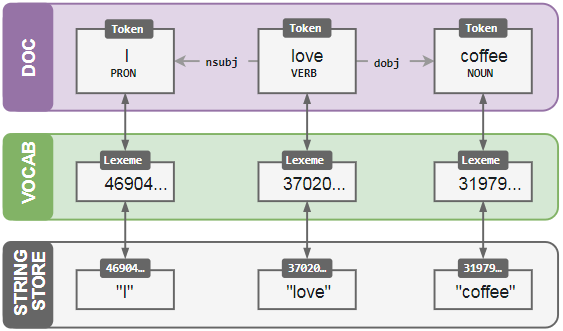
Validation score

What failed? Analysis where the model will not perform correctly

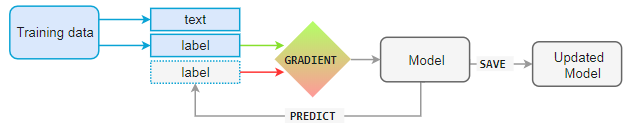
## 4. Discussions

Classification of Documents. Adresses C0

An ensemble model comprising of the bag-of-words model stacked with a neural network model is used for document classification. A convolutional neural network (CNN) along with attention and mean pooling is used in the neural network model. The attention improves the traditional encoder-decoder model used in neural network. Mean pooling is used to limit the over-fitting caused due to the convolved feature. Hash values are assigned for each word in the vocabulary. Hash values help in saving memory, and are also context-independent.



The model is trained with the documents and its corresponding labels. The validation set is used for prediction, and the feedback to the model is given in the form of error gradient of the loss function. The difference between the expected output and the training example is calculated using the error gradient of the loss function. During training the model does not memorise the example, instead it generates a concept that can be generalized over all the training examples. The document classification model is trained for 150 epochs.



# Literature Review

## 1. Deep learning in finance applications

## 2. Loan operations research

# Conclusion

# References

1. Leung MK, Xiong HY, Lee LJ, Frey BJ. Deep learning of the tissue-regulated splicing code. Bioinformat-ics. 2014; 30(12):121–9
2. Ding X, Zhang Y, Liu T, Duan J, editors. Deep learning for event-driven stock prediction. International Conference on Artificial Intelligence; 2015
3. Conference on Artificial Intelligence; 2015
4. Chen AS, Leung MT, Daouk H. Application of neural networks to an emerging financial market: forecasting and trading the Taiwan Stock Index. Computers & Operations Research. 2003; 30(6):901–23.
5. casting and trading the Taiwan Stock Index. Computers & Operations Research. 2003; 30(6):901–23
6. Yue, Jun & Rao, Yulei. (2017). A deep learning framework for financial time series using stacked autoencoders and long-short term memory. PLoS ONE. 12. 10.1371/journal.pone.0180944.
7. Hsieh TJ, Hsiao HF, Yeh WC. Forecasting stock markets using wavelet transforms and recurrent neural networks: An integrated system based on artificial bee colony algorithm. Applied Soft Computing. 2011; 11(2):2510–25
8. Chui, M., Manyika, J., Miremadi, M., Henke, N., Chung, R., Nel, P., & Malhotra, S. (2018). Notes from the AI frontier: Applications and value of deep learning. McKinsey global institute discussion paper, April 2018. Retrieved June 12, 2019 from <https://www.mckinsey.com/featured-insights/artificial-intelligence/notes-from-the-ai-frontier-applications-and-value-of-deep-learning>.
9. Ting (Sophia) Sun (*2019*) Applying Deep Learning to Audit Procedures: An Illustrative Framework. Accounting Horizons: September 2019, Vol. 33, No. 3, pp. 89-109.
10. B. Zhu, W. Yang, H. Wang, Y. Yuan, A hybrid deep learning model for consumer credit scoring, in: International Conference on Artificial Intelligence and Big Data (ICAIBD), 2018, pp. 205–208. doi:10.1109/ICAIBD. 2018.8396195.
11. Neha Gupta, Pratistha Mathur (2012) “Spell Checking Techniques in NLP: A Survey”, International Journal of Advanced Research in Computer Science and Software Engineering, Volume 2, Issue 12, December 2012, pp. 217-221
12. Kibble, R.: Introduction to Natural Language Processing. University of London (2013)
13. J. Jiang Information extraction from text. C.C. Aggarwal, C. Zhai (Eds.), Mining text data, Springer, United States (2012), pp. 11-41
14. Bunkley, Nick (March 3, 2008). "Joseph Juran, 103, Pioneer in Quality Control, Dies". New York Times. Archived from the original on September 6, 2017. Retrieved 25 January 2018.
15. Box, George E.P.; Meyer, R. Daniel (1986). "An Analysis for Unreplicated Fractional Factorials". Technometrics. 28 (1): 11–18. doi:10.1080/00401706.1986.10488093.
16. Marshall, Perry (2013-10-09). "The 80/20 Rule of Sales: How to Find Your Best Customers". Entrepreneur. Retrieved 2018-01-05.
17. Dzmitry Bahdanau, Kyunghyun Cho, and Yoshua Bengio. Neural machine translation by jointly

learning to align and translate. CoRR, abs/1409.0473, 2014.

1. Kyunghyun Cho, Bart van Merrienboer, Caglar Gulcehre, Fethi Bougares, Holger Schwenk,

and Yoshua Bengio. Learning phrase representations using rnn encoder-decoder for statistical

machine translation. CoRR, abs/1406.1078, 2014.

1. Ilya Sutskever, Oriol Vinyals, and Quoc VV Le. Sequence to sequence learning with neural

networks. In Advances in Neural Information Processing Systems, pages 3104–3112, 2014.

1. Graves, Alex. "Generating sequences with recurrent neural networks." *arXiv preprint arXiv:1308.0850* (2013).