National College of Ireland

Estimation of Story Points in Agile Framework Using Deep Learning

MSc Research Project
Data Analytics

Shravan Basawaraj Jewargikar
Student ID: 17154944

School of Computing
National College of Ireland

Supervisor: Dr. Pramod Pathak, Dympna O'Sullivan

National College of Ireland

MSc Project Submission Sheet

School of Computing

Student Name:	Shravan Basawaraj Jewargikar		
Student ID:	x17154944		
Programme:	MSc in Data Analytics	**Year:**	2018-19
Module:	Research Project		
Supervisor:	Dr. Promod Pathak and Dympna O'Sullivan		
Submission Due Date:	20th December 2018		
Project Title:	Estimation of Story Points in Agile Framework Using Deep Learning		
Word Count:	8253	**Page Count:**	27

I hereby certify that the information contained in this (my submission) is information pertaining to research I conducted for this project. All information other than my own contribution will be fully referenced and listed in the relevant bibliography section at the rear of the project.
ALL internet material must be referenced in the bibliography section. Students are required to use the Referencing Standard specified in the report template. To use other author's written or electronic work is illegal (plagiarism) and may result in disciplinary action.

Signature: ……………………………………………………………………………………………………

Date: 20th December 2018

PLEASE READ THE FOLLOWING INSTRUCTIONS AND CHECKLIST

Attach a completed copy of this sheet to each project (including multiple copies)	□
Attach a Moodle submission receipt of the online project submission, to each project (including multiple copies).	□
You must ensure that you retain a HARD COPY of the project, both for your own reference and in case a project is lost or mislaid. It is not sufficient to keep a copy on computer.	□

Assignments that are submitted to the Programme Coordinator Office must be placed into the assignment box located outside the office.

Office Use Only	
Signature:	
Date:	
Penalty Applied (if applicable):	

Table of Contents

Estimation of Story Points in Agile Framework Using Deep Learning Techniques

Shravan Basawaraj Jewargikar

17154944
MSc Research Project in Data Analytics
20th December 2018

Abstract
With the advent of the agile development process, many organizations are drifting from the traditional waterfall development process towards agile methodology. There has been considerable research in effort estimation in the traditional software development process, whereas limited research is carried out for estimation in the agile software development process. The metric which is adapted to identify the complexity in developing the user story or to fix the issue is measured by story points. The dataset comprises of issues which are collected from the 14 open source projects. The study explores experimenting with pre-trained glove 300 dimension embedding method. Alongside the study proposes the estimation model by applying deep learning technique: Bidirectional Recurrent Neural Network (BRNN) with the Gated Recurrent Unit (GRU) as hidden units. In addition to the estimation, the key novelty of the study involves in building a classification model on the story point dataset. This is achieved by applying the same deep learning technique. The story point is categorized into low, medium and high complexity as required for developing the task. The observed evaluation is based on the Mean Absolute Error (MAE) for estimation and percentage accuracy for classification.

Keywords: Agile Development, Story point, Bidirectional recurrent neural network, Mean Absolute Error

1 Introduction

Effort estimation plays an important role in software project management, majorly in the phase of project planning and project supervision. The efficiency of the effort estimation plays a critical role in the output of the software project; overestimation results in organization competitiveness, on other hand underestimation results in the schedule and budget overruns. Software effort estimation can be divided into two different research domains, model-based and expert-based methods. The expert-based method depends on human expertise to make the correct predictions of story points. The model-based method acquires the knowledge from the previous projects to make the predictions of story points for new projects. Most of the research has been conducted in the effort estimation considering the waterfall software development model. With the modernization in the software project management, the Agile methodology has acquired its robust place among the software project management. In the Agile methodology, the software is developed in iterative cycles (*e.g. Sprints in Scrum*) with shorter time frames. This methodology allows ad hoc requirements to be incorporated into the project at any phase of a project's life. Every iteration includes the development of software project requirements, which are termed as the user stories or Issues in Agile. This is the deviation from the traditional software development where all the functionalities are delivered together as a whole to the iterative delivery.

With the advances in adopting the Agile framework in the Software industries, it provides the wide scope for research in the effort estimation. The majority of research is contributed to the estimation of effort considering the development of the whole project which is a traditional approach. With the Agile framework, the estimation of story points for every single iteration (*Sprint in Scrum*) needs the attention of researchers. Every development team goes through the discussion of user stories (Requirement) to estimate the effort of the size which is required to accomplish the task. Story points are the metric which is used to describe the complexity involved in developing the user story. Every iteration (*Sprint*), story pointing starts by assigning the simple task with base value. This base value acts as the standard, which is used in assigning the story points for user stories comparing to the base task. Story points occur in the pattern of Fibonacci series (1, 2, 3, 5, 8, 13, ∞). As the contemporary method for estimation includes the expert opinion, which leads to the different story points for the same piece of the work causing the challenge to the scrum masters and project managers. Presently most of the companies following the Agile methodology for software development rely on methods such as expert opinion, planning poker etc. to conclude with the estimation. These methods incur the human error which results in the overestimation and underestimation.

The proposed research builds the prediction model and classification model which uses the attributes from the dataset to predict and classify the story points. These models are trained using the previous project story points, which will be used to estimate the story points of the new project. This study performs analysis on the dataset which is publicly available (Choetkiertikul *et al.*, 2018). The current study addresses the following research questions,

RQ1. How well the proposed approach is suitable for the estimation of story points?
The objective of the question is to answer, does the proposed model estimates the story points when compared with the base models such as random guessing, mean effort and median effort.

RQ2. How Bidirectional Recurrent Neural Network (BRNN) with Gated Recurrent Unit (GRU) performs in the estimation of story points?
The objective of the question is to build the machine learning model using BRNN which is measured in the metric of Mean Absolute Error in comparison with the base paper (Choetkiertikul *et al.*, 2018).

RQ3. How well the classification of story points is classified into categories (High, Medium, Low) using Bidirectional Recurrent Neural Network (BRNN) with Gated Recurrent Unit (GRU)?
The objective of the question is to build a novel machine learning model for classification of the story points into three categories (High, Medium, Low). As per the best of knowledge from the literature, classification on the story points with Bidirectional RNN is a novel approach with respect to implementation.

Aims and Objectives:

- **Pre-processing of the data**
  The data collected contains the special characters, HTML tags, alphanumeric characters, and extra blank space. The aim of the pre-processing is to get the data to a standard format.
- **Feature Engineering**
  This phase aims to get the features from the textual data. More accurate the features,

higher the accuracy of the model. The model will be trained with Glove embedding methods.

- **Machine Learning Model**
  The research shows that Bidirectional RNN(BRNN) is predominately acquiring the position in text mining for various NLP applications. In this process, the research applies for predicting the story points along with GRU's.
  Classification of story points into three categories high, medium and low which is novel implementation on the story point dataset. This is achieved by the combination of BRNN with GRU hidden units and the pre-trained glove embedding method.
- **Evaluation of the results**
  The results are evaluated based on the metrics Mean Absolute Error (MAE) for prediction and with percentage accuracy for classification. The mentioned metrics are recommended for machine learning models.

The conducted study will be applied in real software industries which follows the agile framework for development. The end users of the application will be a team of software developers, Projects managers, Scrum master, Product owners and business people of the organization. The model can be plugged into JIRA which is used for project management, bug tracking, and issue tracking tool. Along with real-time applications, this study contributes to the research community and motivate the researchers for further contribution in the same domain.

2 Related Work

The existing software development process which is applied in most of the industries is moved from the traditional development approach to the dynamic agile software development process. Indicating the iterative development process with consideration of ad-hoc requirements. Lateral this process gives the wide scope of research in the estimation of story points, which is the metrics of the understanding the complexity of requirements to be developed. An Error in the estimation of story points may lead to adverse effects on the process. There are two types of the approach which is applied in this domain of the estimation of story points: Expert based opinion and Machine learning approach. There are numerous machine learning algorithms which are enforced in predicting story points (Choetkiertikul *et al.*, 2018).

2.1 Study on Agile Development Process

Most of the effort in the project development process is consumed in the planning process, as the planning phase includes the detailed study of the software requirements. The planning phase needs much attention as it contributes to the various aspects of the project, such as product release activities, marketing campaign, and training to the clients concerning the product (Heravi, Coffey and Trigunarsyah, 2015). The traditional project development process shelters many evident disadvantages. For one, half of the resources are consumed at the initial stage of planning (Cervone, 2011). To over these, the agile development process was developed. The agile development process is categorized into different processes such as Extreme project management, dynamic project management, scrum, and adaptive project management. Among all these, the most adopted one is the scrum. According to (Mahnic, 2011), 58% of the respondents voted for the scrum which is the result of the survey conducted as part of a software engineering course.

Agile is the iterative development process which is different from the traditional waterfall model. Every iteration is termed as the sprint in the agile, every sprint has the deliverables which contribute to the whole project. The work which is taken into the sprint is termed as the user stories or issues, every user story answers the combination of three W's (Who, What, Why). The paucity of research in the domain of the estimation of story points cater scope for research. According to(Usman *et al.*, 2014), three crucial drivers for effort estimation 1) skill set of teams; 2) prior knowledge; 3) task/user story size.

2.2 Study on Bayesian network-based effort estimation

Every software industry concentrates on the 'software engineering cost model' which are developed to estimate the cost, quality and the duration of the software. Every machine learning model works on the assumptions before it generates the result, as described in the (Chulani, Boehm and Steece, 1999) states that multiple regression establishes assumptions which breach the software engineering principles of cost models. The results show that the Bayesian model outperforms multiple regression approach considering the performance metric of prediction.

Bayesian Approach: 30% of actual value 75% of the time
Multiple Regression: 30% of actual value 52% of the time

These results are obtained by working on one of the software engineering cost models, COCOMO II.

Project velocity is one of the major metrics in the project development process. The paper illustrates (Hearty *et al.*, 2009) the application of the dynamic Bayesian model in the estimation of project velocity in the Extreme Programming (XP) development process. The study evaluates the learning BN model for XP projects. The result is an output which is acquired by applying the model to the real-time industry project.

2.3 Study on Hybrid Machine learning Models

Estimation of the correct effort required to develop the task is of major concern in the project planning phase. The classifier suggested in (Porru *et al.*, 2016) results in the correct estimation of a new issue in less than 15 seconds. The metrics which are adapted to measure are Magnitude of relative error (MRE) which is between the range of 0.16 and 0.61. The input values for the model was the *Summary and Description* of the user story/task. The steps which are followed in the implementation are,

1. Text preprocessing and cleaning with the removal of stop words.
2. Applying monograms and Bigrams on the concatenated data.
3. Feature extraction: Applied Term Frequency-Inverse document frequency (TF-IDF).
4. The sparse vector matrix is then fed into the classifier for the estimation

Estimator	Correct estimates	Accuracy	MMRE
SVM	413	0.59	0.50
NB	309	0.44	0.85
KNN	249	0.36	0.70
DT	158	0.23	0.98

Table 1: Results from the study (Porru *et al.*, 2016)

With the complement to the traditional approach of the estimation of effort using the 'planning poker' method the study (Moharreri *et al.*, 2016), implements an automated estimation model which is termed as "*Auto Estimate".* Various machine learning justifies this approach where the results depict the decision trees J48 along with planning poker provides the higher accuracy with correct estimation.

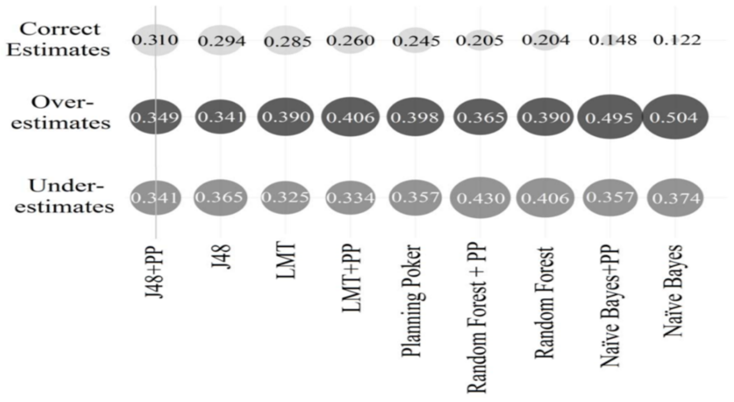


Figure 1: Results from the literature

The metric which is used to evaluate the results are Mean Magnitude of Relative Error (MMRE) where the hybrid of J48 and PP scores the value of 91.75%.

2.4 Study on Word Embedding methods

Almost all of the machine learning model accepts the data to be in the form of vector format. In Natural Language Processing (NLP), word embedding is applied to convert the text data into a vector format. According to the study (Wang *et al.*, 2018), the role of word embedding in the NLP task is to 1) Information retrieval 2) Feature Extraction 3) Sentiment analysis 4) Summarizing of the text. The result of the study indicates that the word embedding which is trained on the corpus of same domain is significant over the pre-trained embedding methods such as Glove and Google News. The experiment results are quite prominent in selecting the word embedding method over the term frequency features as semantics are embedded in the former.

The choice of the word embedding plays a major role in the performance of the model over the architectural choices. The recipe which is described in (Dhingra *et al.*, 2017) for dealing with the problem of answering the questions of *Reading Comprehension* (RC), 1) Words in the corpus must be converted into vectors using word embedding methods such as Glove 2) Sequence model such as Long Short-term Memory (LSTM) converting the vectors into context representation 3) The context representation then determines the answer from the RC. The comparison between Glove and Word2vec, the results were evident with better performer one is Glove. The major difference lies in the different embedding size between them; Glove(50-300d) and Word2vec (300d only).

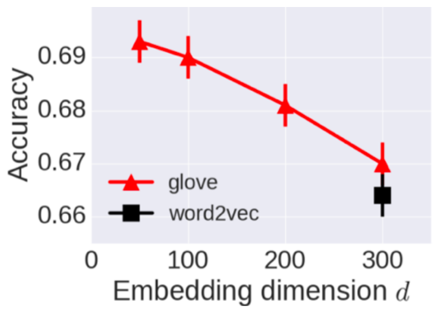


Figure 2: Accuracy Comparison between Glove and Word2vec

With the pre-trained word embedding which is publicly available are the basics of NLP and machine learning models. In the study (Mikolov *et al.*, 2017) Glove Embedding from Stanford NLP group, has proved to be of better results for a large dataset with less model training time. For the options of considering the embedding methods between Glove, fast text, word2vec, and paragram, the most resourceful and are best-untuned embedding (Scheepers, Kanoulas and Gavves, 2018). Depending on the size of the dataset, the embedding methods vary with results, which is discussed in the study (Wieting *et al.*, 2015).

The study (Qi *et al.*, 2018) states the importance of applying the pre-trained embedding on the training corpus has a low frequency of words. The pre-trained embedding methods perform with better accuracy and are widely applied in text classification and sequence tagging. The results are compared using the evaluation metric of F-score

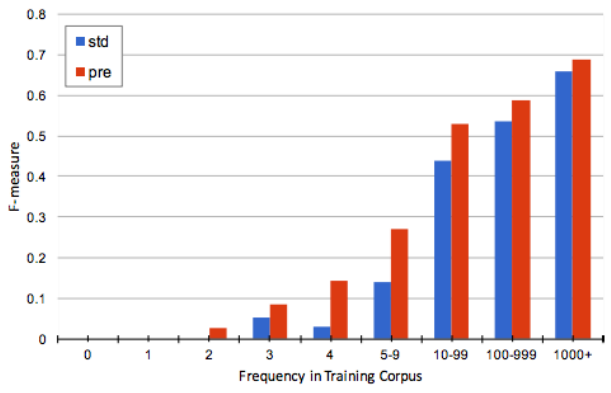
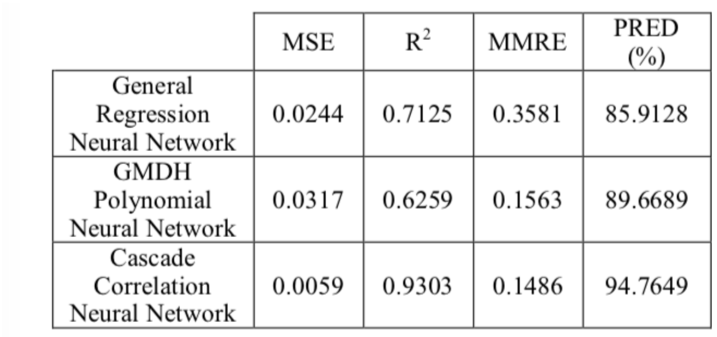


Figure 3: Standard Embedding v/s Pre-trained Embedding

2.5 Study on Neural Network

The organization must be accurate in predicting the story points correctly as that is important in building the project. The study (Panda, Satapathy and Rath, 2015) has adopted and evaluated the Neural Network (NN) models such as General Regression NN (GRNN), Group Method of Data Handling (GMDH) Polynomial NN, and Cascade Correlation NN. The evaluation metric for these models is Mean Magnitude Relative Error (MMRE). Among the mentioned NN, from the results, it can be seen that Cascade Correlation NN outperforms other two NN models.



	MSE	R^2	MMRE	PRED (%)
General Regression Neural Network	0.0244	0.7125	0.3581	85.9128
GMDH Polynomial Neural Network	0.0317	0.6259	0.1563	89.6689
Cascade Correlation Neural Network	0.0059	0.9303	0.1486	94.7649

Table 2: Different Neural Network approach with results

Neural Networks possess its own strengths when it has to be trained on the sequence data. In NLP, the two major used models are Convolution Neural Network (CNN) and Recurrent Neural Network (RNN). As mentioned in (Chen *et al.*, 2016) describes, the pattern recognition from the features are well handled in Bidirectional-RNN's for the prediction in clinical text data. The Bidirectional RNN (BRNN) suits for the data which are dynamic as it performs end to end training with RNN's. Considering the speech recognition data (Graves, Mohamed and Hinton, 2013), BRNN is applied to the study as it contains context understanding from the previous node of the network. The architecture of BRNN which is mentioned in the study is, it consists of the two layers between input and output, i.e., the forward direction and backward direction layers.

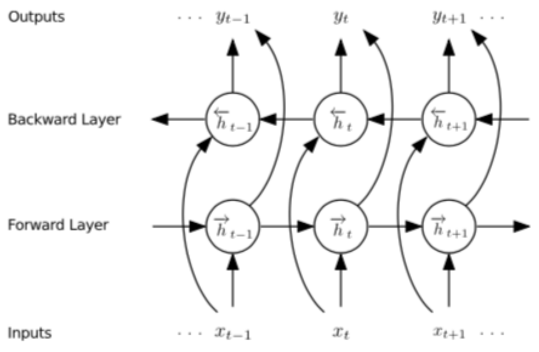


Figure 4: Architecture of Bidirectional RNN

Deep learning is one of the important models which is applied in most of the NLP applications. The study (Badjatiya *et al.*, 2017) conducts the Hate speech detection from the social media platform data. The results indicate that Deep learning works better with a comparison to state-of-the-art char/word n-grams with evaluation metric of the F1 score of ~18 points for deep learning. RNN is proved to be the most powerful for processing sequence data (Hochreiter and Frasconi, 2009). Along with the advantage of RNN's, it comes along with the vanishing gradient problems which make the training difficult for the RNN model

The Bidirectional networks provide more accurate results when compared with unidirectional for the sequence data, where an understanding of the context is of the high importance which contributes to the accuracy. In the study (Gers, Schmidhuber and Cummins, 2000), the bidirectional network is the combination of the probability of the output of the forward net and backward net. The visual graph from the study indicates the advantage of adopting the bidirectional network.

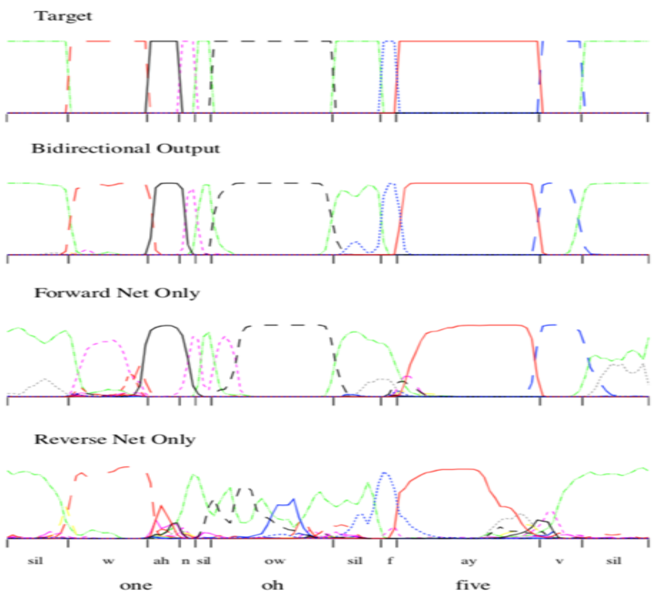


Figure 5: Output similarity between Target and Bidirectional Networks

The figure indicates the classification of the speech from the TIMIT database where the input is *"one oh five"* (Gers, Schmidhuber and Cummins, 2000) where the bidirectional output is similar to the target output. The architecture which is explained in the study (Risch and Krestel, 2018) explains about the advantage of applying Gated recurrent units in the deep learning model for predicting the three categories of aggression. The approach which is followed includes a bidirectional recurrent neural network with a gated recurrent unit with different pooling levels. The evaluated metric which is used in the study is of F1 score, where the study results in an F1 score of 60%. The result is from the RNN's ensemble model.

2.6 Conclusion

In a nutshell, the literature review of various study will help the researcher with the various deep learning methodologies and the advantages it has on the sequence text data. From related studies, the criticality of the problem which is being addressed in the current study is evaluated with its impact on the software development process. As applied architecture in the (Risch and Krestel, 2018) which highlights the importance of bidirectional architecture and gated recurrent units, similar architecture will be applied in the current study. On another hand, for the classification of the story points based on the textual data adopting the methodology from the study (Zhou *et al.*, 2016) on current story point dataset. The results of the machine learning model vary with different embedding methods, as they create the vector data which is fed into the model. The study (Qi *et al.*, 2018) highlights the advantage of pretrained embedding when training process time is considered. The literature review aid in deciding the roadmap for the current project to achieve better performance and also includes the steps to consider before applying to the model.

3 Research Methodology

The methodology describes the phases which are involved in the project and how every phase is connected to achieve the result in a most efficient way. Lately, many types of research have shown the importance of Cross Industry Standard Process Data Mining (Crisp-DM) in text mining (Olmer, 2001). For the current study where development is follows the iterative process. In this case, Crisp-DM is well suitable for the project.

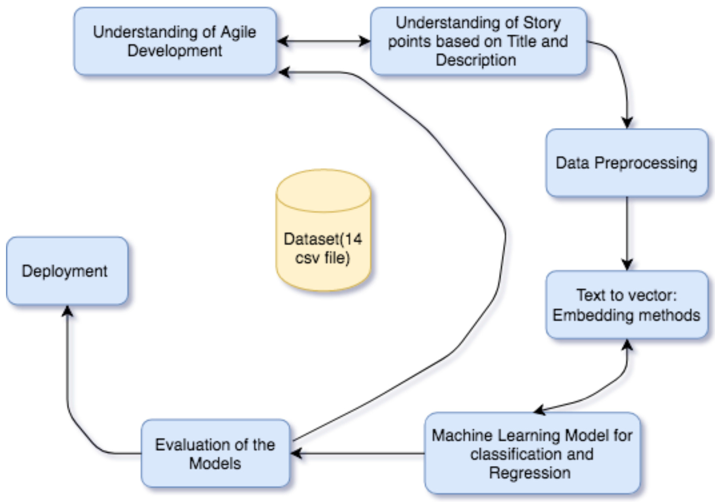

Understanding of Agile Development
Understanding of Story points based on Title and Description
Data Preprocessing
Dataset(14 csv file)
Deployment
Text to vector: Embedding methods
Machine Learning Model for classification and Regression
Evaluation of the Models

Figure 6: Proposed Methodology CRISP-DM

Every step indicates the flow of the project starting from business understanding to deployment. Recently because of the dynamic nature of Agile development process many industries are driving the process towards agile. The first phase includes the understanding of story points, complexity involved in estimation of story points, how the estimation of story points contributes to the performance of the organization. The study includes the estimation of story points as the target variable with title and description as the independent variables. With the presence of the data in the text format, the preprocessing and cleaning of the data is the most crucial step in the overall performance of the model. Every machine learning model excepts the data in the numerical format, so the next phase takes care of converting the data into a vector format. In this, every word is represented with the vectors indicating its position in the vector space. The vector data is then fed into the machine learning models, which is followed by an evaluation of the results.

4 Design Specification

For the current proposed study, the architecture which is applied is the two-tier architecture. Where the tier 1 indicates the database layer, which contains all CSV files of 14 projects. The data which is present is in the raw format is then passed to the next layer of architecture. The second tier contains the python Ide: Jupyter notebook. The client layer is used for analysis by applying the measures from the statistics to find the performance of the model. The anaconda navigator which is open source distribution package management and deployment tool for python. It contains many IDEs among which the study is carried out using jupyter notebook. The client layer then presents with the results from the analysis. As the study limits with the time scale, the two-tier architecture is suitable where the client layer includes the evaluation of the machine learning models. The libraries which are used in the study includes the pandas for data manipulation, keras which uses tensorflow backend for deep learning algorithms, nltk library for handling the natural language processing task, and genism for visualization with the word vectors.

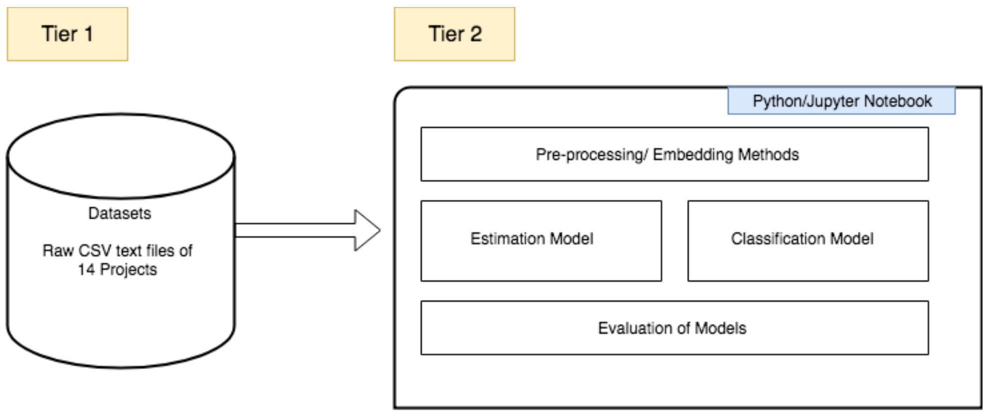
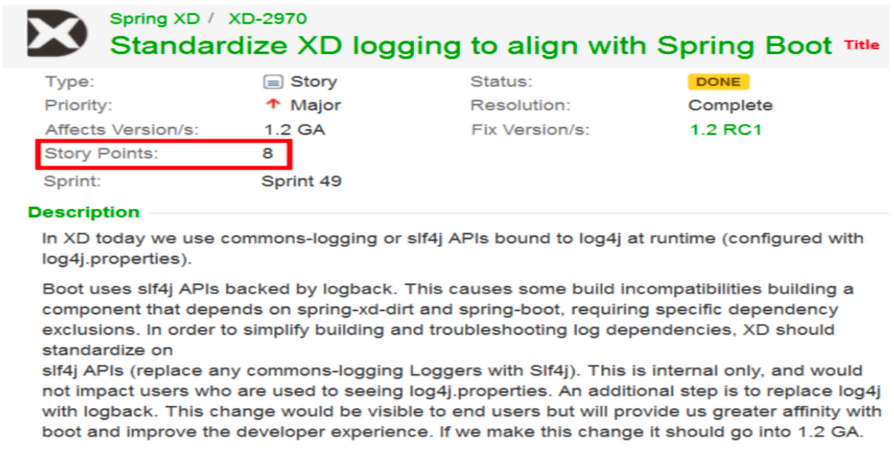


Figure 7: Two-tier architecture for proposed study

5 Dataset

There are many datasets which are available publicly and are valuable assets to the research community for software effort estimation in the project level. From the study (Choetkiertikul *et al.*, 2018), it is noted that story point dataset[1] which is made available publicly consist of 9 open repository projects which use JIRA as a tracking tool. The repositories, in turn, consist of 14 projects. This provides diversity in the dataset with respect to the application domain, knowledge and different characteristics of the project. Each CSV file consists of the attributes, IssueID, Title, Description and Story points, where IssueID is the unique value for the issues/stories in the CSV file, the title which indicates the abstract of what has to be developed as part of the requirement and Description which indicates in brief about what and why the feature has to be developed.



Spring XD / XD-2970

Standardize XD logging to align with Spring Boot Title

Type:	Story	Status:	DONE
Priority:	Major	Resolution:	Complete
Affects Version/s:	1.2 GA	Fix Version/s:	1.2 RC1
Story Points:	8		
Sprint:	Sprint 49		

Description

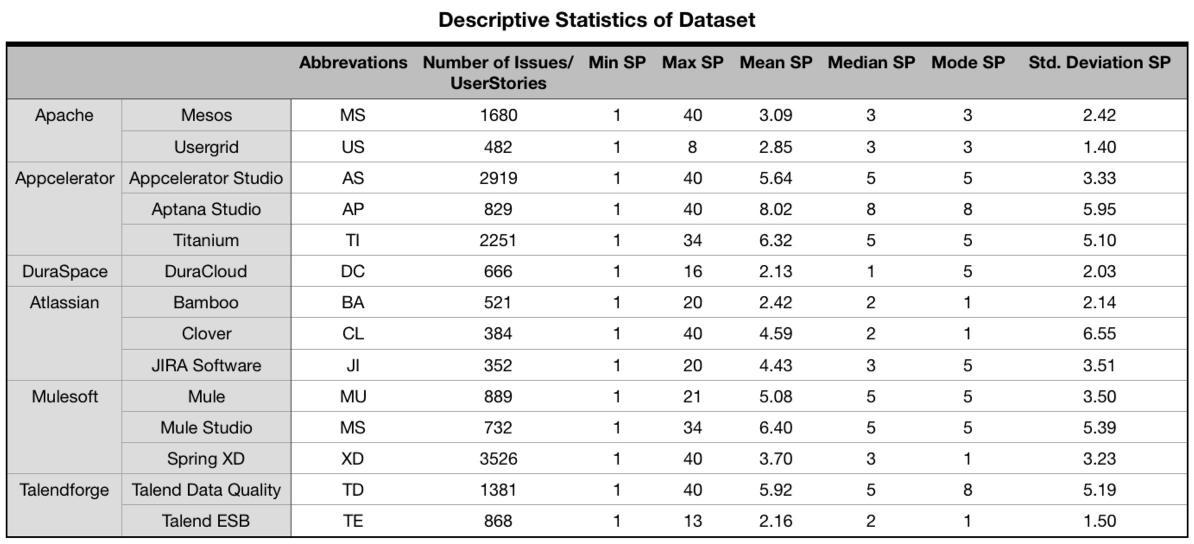
In XD today we use commons-logging or slf4j APIs bound to log4j at runtime (configured with log4j.properties).

Boot uses slf4j APIs backed by logback. This causes some build incompatibilities building a component that depends on spring-xd-dirt and spring-boot, requiring specific dependency exclusions. In order to simplify building and troubleshooting log dependencies, XD should standardize on
slf4j APIs (replace any commons-logging Loggers with Slf4j). This is internal only, and would not impact users who are used to seeing log4j.properties. An additional step is to replace log4j with logback. This change would be visible to end users but will provide us greater affinity with boot and improve the developer experience. If we make this change it should go into 1.2 GA.

Figure 8: Snapshot of story from SpringXD project

[1] https://github.com/SEAnalytics/datasets/tree/master/storypoint/IEEE%20TSE2018/dataset

The snapshot depicts the estimation process with the title and description. The description states the steps, methods or the process which needs to be developed. Every user story addresses the three question for *what* the requirement has to be developed, *who* will be benefited with the development and *why* it has to be developed.


Descriptive Statistics of Dataset

		Abbrevations	Number of Issues/ UserStories	Min SP	Max SP	Mean SP	Median SP	Mode SP	Std. Deviation SP
Apache	Mesos	MS	1680	1	40	3.09	3	3	2.42
	Usergrid	US	482	1	8	2.85	3	3	1.40
Appcelerator	Appcelerator Studio	AS	2919	1	40	5.64	5	5	3.33
	Aptana Studio	AP	829	1	40	8.02	8	8	5.95
	Titanium	TI	2251	1	34	6.32	5	5	5.10
DuraSpace	DuraCloud	DC	666	1	16	2.13	1	5	2.03
Atlassian	Bamboo	BA	521	1	20	2.42	2	1	2.14
	Clover	CL	384	1	40	4.59	2	1	6.55
	JIRA Software	JI	352	1	20	4.43	3	5	3.51
Mulesoft	Mule	MU	889	1	21	5.08	5	5	3.50
	Mule Studio	MS	732	1	34	6.40	5	5	5.39
	Spring XD	XD	3526	1	40	3.70	3	1	3.23
Talendforge	Talend Data Quality	TD	1381	1	40	5.92	5	8	5.19
	Talend ESB	TE	868	1	13	2.16	2	1	1.50

Table 3: Dataset Description

The table shows the statistical details of the dataset which consist of 14 projects from 9 open source repositories. The details follow with the highlighting the numbers of issues which is present in each file along with the range of story points present. The mean value of story points represents the average score of complexity involved in the respective projects. Similarly, the median value indicates the middle value in the story point attribute and mode being the most repeated story point value in the respective project. The standard deviation of the target variable highlights the quantity value which indicates the dispersion of values in comparison with the mean value. These values are the basics of statistics and are the very first step in understanding the data.

6 Implementation

The project aims in building the classification model for categorizing the story points into high, medium and low categories. In addition to this, the project also aims to build a better estimation model with comparison to the base paper (Choetkiertikul *et al.*, 2018). In any issue, tracking tool title and description are the basic requirements when creating an issue. In some other tools (eg. JIRA), there is provision to add other information for the issue like the priority of the task, teams to be assigned to the task, impact on the other teams and so on; but these all are not always mentioned when the issue is being created. Hence in this study, we will have the bleak assumption of relying only on title and description of the issue for the estimation of story points.

As we have different projects from various domains with different characteristics, the training data and testing data should belong to a same CSV file which is the same project. Hence, the approach which is applied in the study is to perform the estimation and classification on individual projects. Training the data and testing the data from the same project will be the appropriate approach. At the first instance, the preprocessing of data includes the huge work which concentrates on removing the noise from the data. With the help of regular expression functions from *"re"* library we have achieved to remove the unwanted HTML tags, null values, HTML links, punctuations, and stop words. The words with the length less than 2 and which are not in the stop word list are also removed as they don't contribute into the model prediction. As we have the sequence of words which is used as input, Bidirectional Recurrent Neural Network is applied in the approach for classification and regression (Arisoy *et al.*, 2015).

For the machine learning model to process the data and perform the estimations, the input data must be in the numerical format. In this study as there is text data, before pushing the data to the model it has to be converted into a vector format. In this study, we will be applying pre-trained embedding method[2]: Glove embedding method with 300 dimension vector value. The design of the approach is divided into four components: 1. Word Embedding 2. Gated Recurrent unit 3. Bidirectional RNN 4.Regressor.

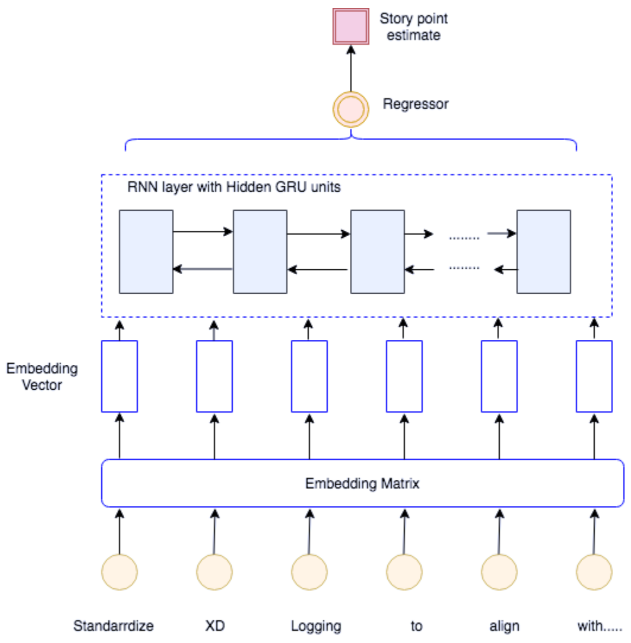


Figure 9: Proposed Estimation Model

The figure shows the sequence of words which is fed into the embedding matrix of the pre-trained embedding methods. Every single word is then represented with high dimensional space vector values. The GRU units are applied in the architecture which avoids the vanishing gradient problem with RNN units. As the data consist of long sequence data, where it will be difficult to keep the data in the memory for the longer run. The bidirectional RNN along with GRU units captures the context and semantics along with solving vanishing gradient problems of RNN. With the final "*relu"* activation function the story point is estimated. For every individual project, the data is split into training and testing with 70% and 30% before applying to the models. The above approach can be summarized as,

Output = Regressor (BRNN (GRU (Embedding Method(s))))

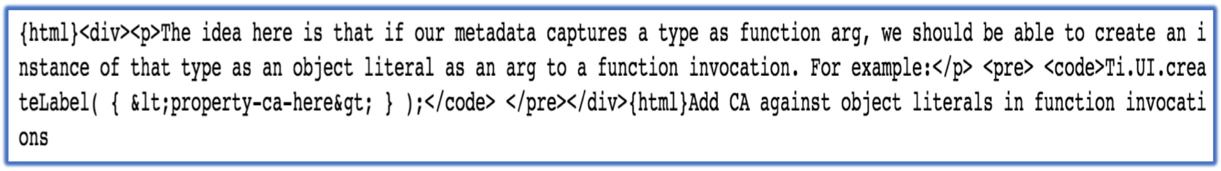
Where s indicates the sequence of words, "*Standardize XD Logging to align with… "*

[2] https://nlp.stanford.edu/projects/glove/

6.1 Preprocessing of the data

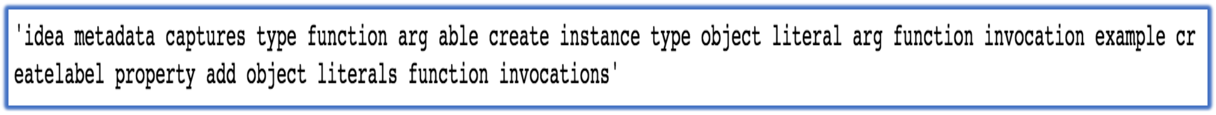
The implementation starts with the cleaning of the data and getting the data into a standard format. As the data is human written language format so the data contains the noise which covers human errors, technical explanations, HTML tags, and HTML links. Removing the values which do not hold any important information will enhance the performance of the model. For preprocessing, the library 're' is applied which uses a regular expression to identify the unwanted patterns in the data. The title and description are merged together into the single column, as both the attributes contribute to the prediction and classification of story points.

Before Preprocessing:



```
{html}<div><p>The idea here is that if our metadata captures a type as function arg, we should be able to create an i
nstance of that type as an object literal as an arg to a function invocation. For example:</p> <pre> <code>Ti.UI.crea
teLabel( { &lt;property-ca-here&gt; } );</code> </pre></div>{html}Add CA against object literals in function invocati
ons
```

After Preprocessing:



```
'idea metadata captures type function arg able create instance type object literal arg function invocation example cr
eatelabel property add object literals function invocations'
```

The estimation of story points is based on every individual project, as the context and project characteristics changes which will impact the final estimation of story points. Hence every project's data is preprocessed, and the data is split into training and test data with 70:30 ratio. After the split between train and test, words are converted into tokens which is achieved by applying the *tokenizer* function. This step helps in transforming the unstructured data into more usable data, which is easy for further enhancements.

6.2 Glove Embedding Method

When the data cleaning is achieved and has the data in the standard format, the data than must be converted into the numerical format which is required by the machine learning models. For the conversion of the text to numerical, the pre-trained glove embedding method is applied (Qi *et al.*, 2018). The training data contains the words with a low-frequency count which advice for applying the pre-trained embedding method. The below graph is from the project springXD, which highlights the frequency count of words on y-axis and x-axis contain the words which are present in the data.

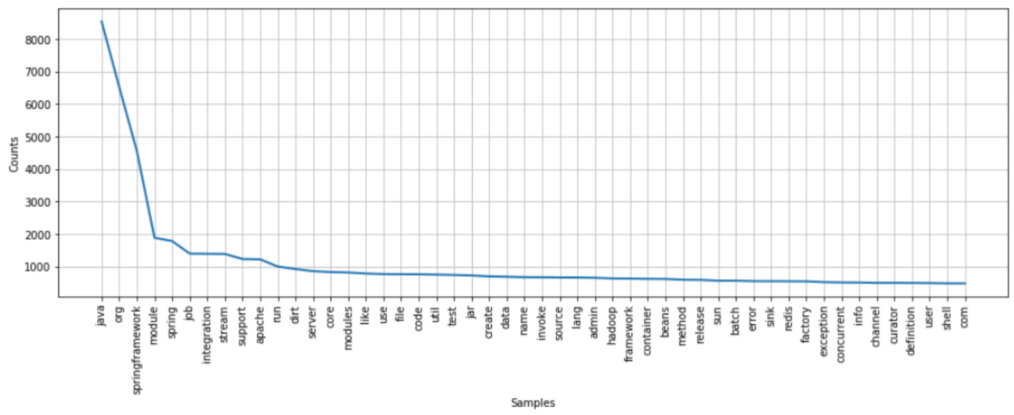
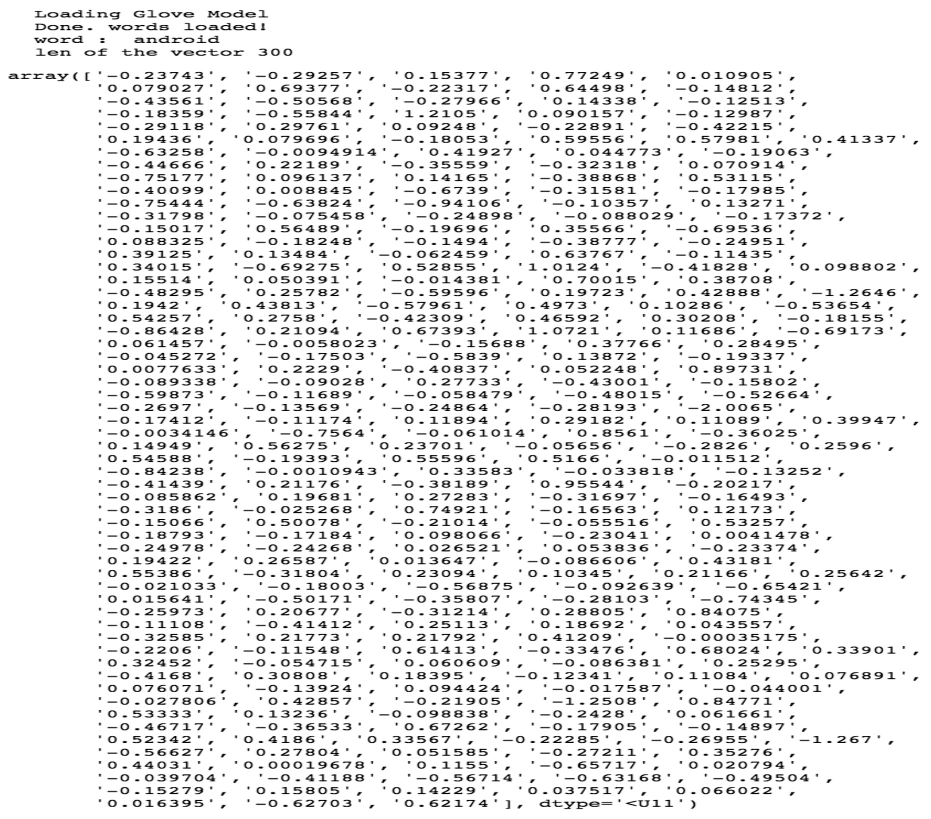


Figure 10: Frequency of words in SpringXD project

The glove embedding method learns by constructing the co-occurrence matrix, which indicates the frequency of words appearing in the context. This results in a huge matrix, hence the factorization on the matrix is carried out to get the low dimension vector values. The figure is of the embedding vector value for the word "*android*" with the length of the vector as 300.



```
    Loading Glove Model
    Done. words loaded!
    word :  android
    len of the vector 300

array(['-0.23743', '-0.29257', '0.15377', '0.77249', '0.010905',
       '0.079027', '0.69377', '-0.22317', '0.64498', '-0.14812',
       '-0.43561', '-0.50568', '-0.27966', '0.14338', '-0.12513',
       '-0.18359', '-0.55844', '1.2105', '0.090157', '-0.12987',
       '-0.29118', '0.29761', '0.09248', '-0.22891', '-0.42215',
       '0.19436', '0.079696', '-0.18053', '0.59556', '0.57981', '0.41337',
       '-0.63258', '-0.0094914', '0.41927', '0.044773', '-0.19063',
       '-0.44666', '0.22189', '-0.35559', '-0.32318', '0.070914',
       '-0.75177', '0.096137', '0.14165', '-0.38868', '0.53115',
       '-0.40099', '0.008845', '-0.6739', '-0.31581', '-0.17985',
       '-0.75444', '-0.63824', '-0.94106', '-0.10357', '0.13271',
       '-0.31798', '-0.075458', '-0.24898', '-0.088029', '-0.17372',
       '-0.15017', '0.56489', '-0.19696', '0.35566', '-0.69536',
       '0.088325', '-0.18248', '-0.1494', '-0.38777', '-0.24951',
       '0.39125', '0.13484', '-0.062459', '0.63767', '-0.11435',
       '0.34015', '-0.69275', '0.52855', '1.0124', '-0.41828', '0.098802',
       '0.15514', '0.050391', '-0.014381', '0.70015', '0.38708',
       '-0.48295', '0.25782', '-0.59596', '0.19723', '0.42888', '-1.2646',
       '0.1942', '0.43813', '-0.57961', '0.4973', '0.10286', '-0.53654',
       '0.54257', '0.2758', '-0.42309', '0.46592', '0.30208', '-0.18155',
       '-0.86428', '0.21094', '0.67393', '1.0721', '0.11686', '-0.69173',
       '0.061457', '-0.0058023', '-0.15688', '0.37766', '0.28495',
       '-0.045272', '-0.17503', '-0.5839', '0.13872', '-0.19337',
       '0.0077633', '0.2229', '-0.40837', '0.052248', '0.89731',
       '-0.089338', '-0.09028', '0.27733', '-0.43001', '-0.15802',
       '-0.59873', '-0.11689', '-0.058479', '-0.48015', '-0.52664',
       '-0.2697', '-0.13569', '-0.24864', '-0.28193', '-2.0065',
       '-0.17412', '-0.11174', '0.11894', '0.29182', '0.11089', '0.39947',
       '-0.0034146', '-0.7564', '-0.061014', '0.8561', '-0.36025',
       '0.14949', '0.56275', '0.23701', '-0.05656', '-0.2826', '0.2596',
       '0.54588', '-0.19393', '0.55596', '0.5166', '-0.011512',
       '-0.84238', '-0.0010943', '0.33583', '-0.033818', '-0.13252',
       '-0.41439', '0.21176', '-0.38189', '0.95544', '-0.20217',
       '-0.085862', '0.19681', '0.27283', '-0.31697', '-0.16493',
       '-0.3186', '-0.025268', '0.74921', '-0.16563', '0.12173',
       '-0.15066', '0.50078', '-0.21014', '-0.055516', '0.53257',
       '-0.18793', '-0.17184', '0.098066', '-0.23041', '0.0041478',
       '-0.24978', '-0.24268', '0.026521', '0.053836', '-0.23374',
       '0.19422', '0.26587', '0.013647', '-0.086606', '0.43181',
       '0.55386', '-0.31804', '0.23094', '0.10345', '0.21166', '0.25642',
       '-0.021033', '-0.18003', '-0.56875', '-0.092639', '-0.65421',
       '0.015641', '-0.50171', '-0.35807', '-0.28103', '-0.74345',
       '-0.25973', '0.20677', '-0.31214', '0.28805', '0.84075',
       '-0.11108', '-0.41412', '0.25113', '0.18692', '0.043557',
       '-0.32585', '0.21773', '0.21792', '0.41209', '-0.00035175',
       '-0.2206', '-0.11548', '0.61413', '-0.33476', '0.68024', '0.33901',
       '0.32452', '-0.054715', '0.060609', '-0.086381', '0.25295',
       '-0.4168', '0.30808', '0.18395', '-0.12341', '0.11084', '0.076891',
       '0.076071', '-0.13924', '0.094424', '-0.017587', '-0.044001',
       '-0.027806', '0.42857', '-0.21905', '-1.2508', '0.84771',
       '0.53333', '0.13236', '-0.098838', '-0.2428', '0.061661',
       '-0.46717', '-0.36533', '0.67262', '-0.17905', '-0.14897',
       '0.52342', '0.4186', '0.33567', '-0.22285', '-0.26955', '-1.267',
       '-0.56627', '0.27804', '0.051585', '-0.27211', '0.35276',
       '0.44031', '0.00019678', '0.1155', '-0.65717', '0.020794',
       '-0.039704', '-0.41188', '-0.56714', '-0.63168', '-0.49504',
       '-0.15279', '0.15805', '0.14229', '0.037517', '0.066022',
       '0.016395', '-0.62703', '0.62174'], dtype='<U11')
```

Figure 11: Word "Android" in 300-dimension vector space

The word "*android"* creates the array of vector values which helps in the identification of the word with 300-dimension values.

6.3 Estimation Model

The basic concept of the design is the Recurrent Neural Network (RNN), which overcomes the issues from the feed-forward network and convolutional neural networks when the input for the model is sequence data. In the feed forward network input is expected to be of fixed length, whereas the sentences vary by length. Hence this arises the problems with feed-forward, though padding the input with fixed size may solve the problem, but still, they perform poorly in comparison with RNN. In the current implementation, the sequencing process of RNN is "many to one". As the input data is the title and description (*many*) and the expected output is the single predicted value of story point (*one*). The basic architecture of RNN is as shown in the figure,

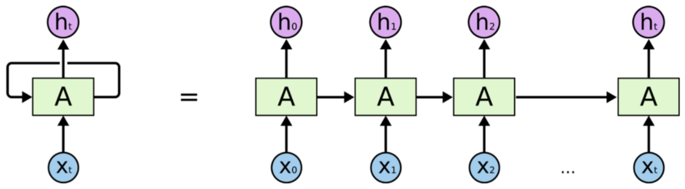


Figure 12: General Architecture of RNN

At every time stamp, the node takes the input x_t (current input) and a_t (output from the previous node) to produce h_t (current output). This process carries out until all timestamps are evaluated. The formula for the current state is written as,

$$A_t = f(h_{t-1}, x_t)$$

Now simple *tanh* is considered as the activation function. Along with the weight matrixes. So, the formula:

$$A_t = tanh\ (W_{hh}.\ h_{t-1} + W_{xh}.\ x_t)$$

Where, tanh: simple activation function, W_{hh} is weight matrix and W_{xh} weight matrix of the current input.

Finally, the output is calculated with:

$$h_t = W_{hy}.A_t$$

In the conventional RNN, the model learns from the data with only previous timestamps, which may not be accurate in some scenarios. This issue is solved with Bidirectional Recurrent Neural Network (BRNN). Consider the example below, with two sentences *S1* and *S2*:

S1: “He said, Apple fruit is essential.” *S2*: “He said, Apple phone is on the table.”

In both the sentences, if the model wants to predict the next word of “Apple” just by using the input as “*He said,* “creates the ambiguity between the fruit and phone. As a result, sometimes it is better to learn from the future representation to eliminate the ambiguity. The combination of forwarding and backward direction resolves the problem. BRNN along with Gated Recurrent Unit (GRU) performs the with high accuracy. The GRU unit in the BRNN solves the vanishing gradient problems (Pascanu, Mikolov and Bengio, 2013) which occur in the conventional RNN. The GRU contains gate’s concept to solve the vanishing gradient problem. It contains two gates 1. Update gate and 2. Reset gate. The architecture of GRU is:

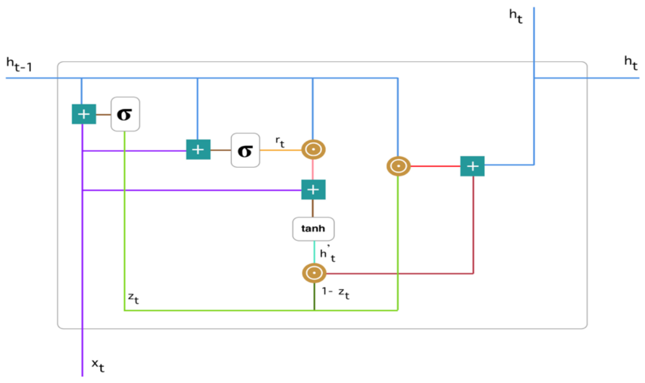


Figure 13: Architecture of Gated Recurrent Unit

Update gate: It helps the model to determine how much of past information must be passed along the future. The formula which indicates update gate is:

$$z_t = \sigma\,(W^z .\ x_t + U^z.h_{t-1})$$

where σ is the activation function which is applied to squash the results between 0 and 1. And W and U indicate the weights respectively.

Reset gate: This gate helps the model to decide how much of the information it must forget so that it removes the irrelevant information to improve accuracy.

$$r_t = \sigma\,(W^r.x_t + U^{rh}.\ h_{t-1})$$

The novel approach includes the BRNN with GRU for predicting the story points as it shows better performance in comparison to LSTM (Xiong, Merity and Socher, 2016). With the help of the keras[3] which uses tensorflow back-end, sklearn, nltk and pandas the implementation is achieved. The output from the glove embedding method which is the vector format is fed into the GRU blocks in the BRNN architecture. The activation function "*relu*", it is applied in the middle of the network to regularize the activation. This function does not suffer from the vanishing gradient problem. The final layer activation function is also "*relu*", which is treated as the universal approximator and contributes in the prediction of story points. The model is run with different dense layers to find the efficient value for the same. From all the 14 projects with the help of the graphical representation, dense layer value with 32 provides with least mean absolute error for the model. The below graph is one of the project outputs from the 14 projects. The loss function of which is used in the implementation is Mean Squared Error (MSE) is widely applied in the linear regression to measure the performance. The evaluation metric for the prediction model is Mean Absolute Error (MAE) (Choetkiertikul *et al.*, 2018). For the current model the MAE for all 14 CSV files ranges between 0 to 4.2 with dense layer equals 32, indicating the least error difference between the actual and the predicted values. MAE is calculated with formula,

$$\text{MAE} = \boxed{\frac{1}{N}\sum_{i=1}^{N}|Actual\ SP - Predicted\ SP|}$$

Where, N: the number of issues present in the file.

[3] https://keras.io/

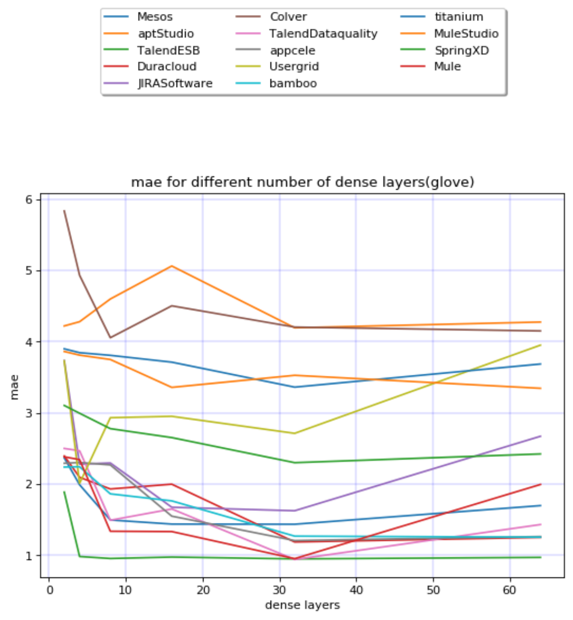


Figure 14: Result of 14 projects with different dense layers

6.4 Classification Model

6.4.1 One Hot Encoding

For the classification model, the novel approach on the current Story Point (SP) dataset is achieved by creating three class labels: *High, Medium and Low.* These class labels indicate the complexity which is involved in the developing the issue or user story. The target variable, story point, which lies in the range of 1- 40 across all 14 projects is converted into binary vectors. This process is achieved using one-hot encoding. The class labels are categorized as,

Low	SP <= 10
Medium	SP >10 and SP <=20
High	SP >20

Table 4: Categories of Story points

After the successful encoding of SP, the column is dropped. In place of that, three attributes are generated. The below figure indicates the results of one-hot encoding, where the column of SP is replaced with three additional columns indicating class label values.

issuekey	title	description	comment_text	high	low	medium
TIMOB-8075	android debugger running cannot back back app	debug android app back app back hangs splash s...	debug android app back app back hangs splash s...	0	1	0
TIMOB-559	android appversion never taken tiapp xml	found bug created new release android market a...	found bug created new release android market a...	0	1	0
TIMOB-684	android border properties broken imageview	code android border around image visible tried...	code android border around image visible tried...	0	1	0

Table 5: Result of One-hot encoding

6.4.2 Classification Model

The classification model brings the novelty to research to the best of knowledge from the literature. The majority of the data is biased toward low complex, as the data is from the real-world projects. The SP are majorly distributed in the range of 0-10 SP value. In the real time scenario, the task which is committed to finish in one sprint (*generally 10 days equals 1 sprint in scrum)* comprises a SP less than 8. If the task exceeds the 8 SP then the task is break down into two different tasks. Hence due to this we have imbalanced data, which is not handled, assuming the real-world situation fits the dataset. The architecture of the proposed model is,

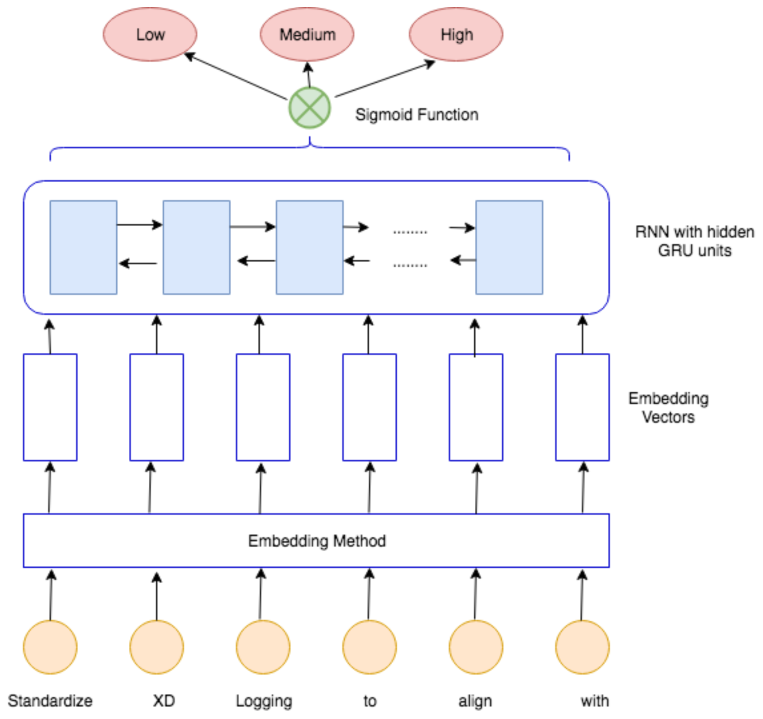


Figure 15: Proposed Classification Model

The implementation follows the same guidelines as of the estimation model, indicating the GRU along with BRNN contribute in the classification results. In this case, the activation function plays the major role in achieving the final probability. The BRNN along with GRU hidden units is applied to achieve the results. The dropout function layer is applied which losses 10% of the data avoiding the overfitting of the model. The *"relu"* activation in the middle of the layer is applied to regularize the activation. The final *"sigmoid"* function in the last layer identifies the probability of classifying the predicted value. This function plays the major role in the classification model as the output of the function ranges between (0,1). The below graph indicates the function of sigmoid and the equation associated with it.

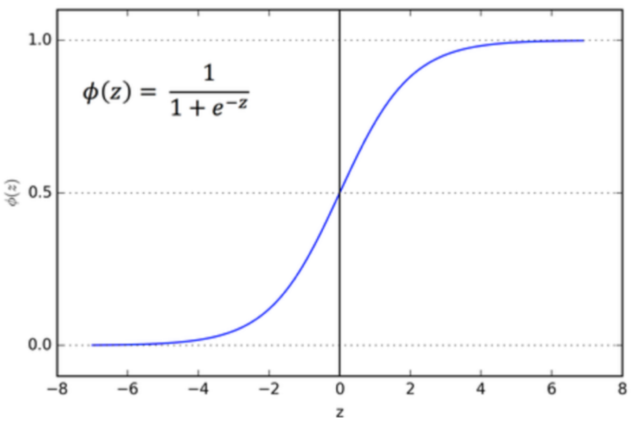


Figure 16: Sigmoid function

The Sigmoid curve indicates the values range between (0,1) and the one hot encoded value created the sparse values, indicating 1 with being present in the class label and 0 as being not present in the class label. With the classification model the results which are achieved in for all the 14 CSV files ranges between the 92% to 98%.

7 Evaluation

The experimental evaluation is carried out to answer the mentioned research questions. The applied metric is with the consideration from the literature (Choetkiertikul *et al.*, 2018) which is considered as base paper for comparison.

RQ1: Suitability check: *How well the proposed approach is suitable for the estimation of story points?*
The results are compared with the base paper (Choetkiertikul *et al.*, 2018) results. In the base paper the comparison is made with respect to random, median and mean estimates. The current approach provides the better results in 5 projects out of 14 projects with the evaluation metric as MAE. This ratio determines that the current approach does not provide better results with comparison to the base paper.

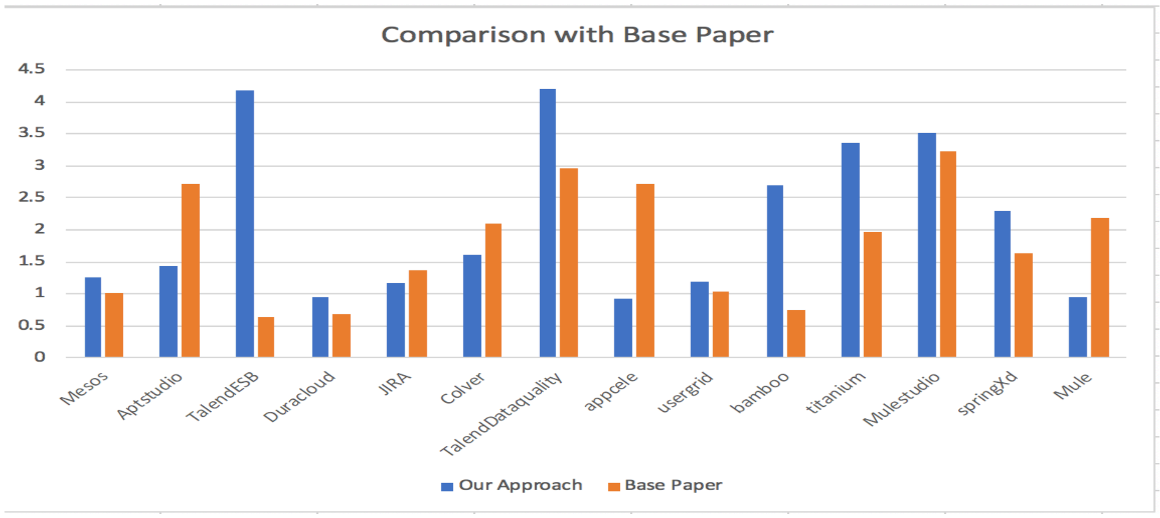


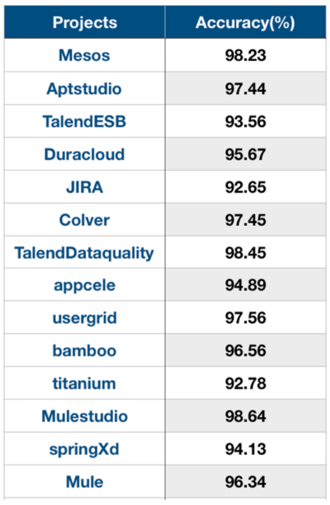
Figure 17: Comparison of proposed approach v/s base paper

RQ2: Estimation Model: *How Bidirectional Recurrent Neural Network (BRNN) with Gated Recurrent Unit (GRU) performs in the estimation of story points?*
The model is evaluated using the MAE metric. The estimation model along with pre-trained glove embedding method results with the values ranging from the 0 to 4.2. MAE is the negative oriented measure, hence lower the value better the accuracy. This range of results is considerably good for the estimation of story points. The below table replicates the output of the estimation model with 32 dense layers. The lower the value of MAE, better the results.

Projects	MAE
Mesos	1.26
Aptstudio	1.43
TalendESB	4.19
Duracloud	0.94
JIRA	1.18
Colver	1.62
TalendDataquality	4.2
appcele	0.93
usergrid	1.2
bamboo	2.7
titanium	3.36
Mulestudio	3.52
springXd	2.3
Mule	0.95

Table 6: Results of Estimation Model

RQ3: Classification Model: *How well the classification of story points is classified into categories (High, Medium, Low) using Bidirectional Recurrent Neural Network (BRNN) with Gated Recurrent Unit (GRU)?*
The classification model on the story point dataset is the novel approach which is covered in the current study, as from the literature there are no evidences about classification work which is performed on the dataset. The percentage metric highlights how accurate the classification of stories/issues into the mentioned categories.



Projects	Accuracy(%)
Mesos	98.23
Aptstudio	97.44
TalendESB	93.56
Duracloud	95.67
JIRA	92.65
Colver	97.45
TalendDataquality	98.45
appcele	94.89
usergrid	97.56
bamboo	96.56
titanium	92.78
Mulestudio	98.64
springXd	94.13
Mule	96.34

Table 7: Results of Classification Model

The results highlight with great accuracy as most of the data present was falling into the low category. This occurs even in the real world, where the issues with large value SP are broken into different task for the convenience of the software developers. Hence, this model provides accurate results across all projects.

7.1 Discussion

The project deals with the estimation of story points as the first model and classification of story points into three categories: high, medium and low as the second model. The pre-trained glove embedding method contributes to the conversion of text data into numerical format. For every machine learning model, more the data better the performance of the model. In the current project, the data which is collected from the individual project from the open source repositories provides very fewer data. For the better results, considering more updated data from the open source projects helps in gaining better accuracy. With the data being in large volume helps in building own standard embedding methods over pre-trained embedding method. There is very limited research carried out on the current dataset which is related to the estimation of story points at the issue level of the project, resulting in a greater volume of noise in the dataset. Further enhancements in the preprocessing considering all the projects will vary the final output.

The classification model is the key novelty in the project. To the best of knowledge, classification has not been performed on an issue/story level dataset. The story point ranges from 1 to 40, whereby considering the real-world scenario the larger volume of data is more biased to fall in the category of low which SP less than or equal 10. This model will be suitable for classifying correctly when the data volume increases in the other two categories: medium and high. Applying the under sampling may lead to losing the information, which is concerned with the majority class. The estimation model is performing with the good results where MAE value is ranging with 0 to 4, which implies in the good and correct estimation of story points with the least error between the actual and predicted values. When the values are compared with the base paper, the embedding methods causes the differences where the word2vec in base papers is performing better compared to the pre-trained glove embedding with 300 dimensions in the current project.

8 Conclusion and Future Work

In summary, the study proposes the deep learning model for story point estimation with pre-trained glove word embedding method. The combination of BRNN with GRU hidden units performs better with MAE values as shown in the evaluation section of the report. However, with the comparison to the base paper, the proposed approach outperforms the base paper results in 5 projects among 14 projects. The key novelty in this study comprises of a classification model with BRNN and GRU hidden units on the story point dataset. The classification labels are low, medium and high. The result for the classification model is with an average of 95% for all 14 projects. The results of the classification model help the end user to identify the category by understanding the complexity involved in developing it. Along with the implementation of models, this study contributes to the research community of Agile and addressing the criticality of story point estimation.

The future work involves extracting more data from open source project repositories which provides better performance of models. With the addition of other attributes such as the priority of the task, team expertise results in better classification and prediction. In addition, experimenting with different embedding method such as fasttext will also be crucial. Furthermore, the classification of story points can be explored using long-short-term memory with a convolution neural network. Lastly incorporating the model into real practice, building the model in issue tracking tool such as JIRA. This helps in evaluating the model and its benefits in real practices.

9 Acknowledgment

I would like to express my gratitude towards my supervisors Dympna O' Sullivan for continuously supporting in thesis implementation and writing which acted as a strong base for me to execute my project. The valuable feedback and review in the thesis classes helped me to get focused in right direction. And also, I thank Dr. Promod Pathak for his continuous motivation and encouragement to achieve what has been planned as part of the project. I would like to thank my parents for their continuous support and care throughout my research. Finally, I thank my friends for continuous moral support during my research.

References

Arisoy, E. *et al.* (2015) 'Bidirectional recurrent neural network language models for automatic speech recognition', *ICASSP, IEEE International Conference on Acoustics, Speech and Signal Processing - Proceedings*. IEEE, 2015–Augus, pp. 5421–5425. doi: 10.1109/ICASSP.2015.7179007.

Badjatiya, P. *et al.* (2017) 'Deep Learning for Hate Speech Detection in Tweets', (2). doi: 10.1145/3041021.3054223.

Cervone, H. F. (2011) 'Understanding agile project management methods using Scrum', *OCLC Systems and Services*, 27(1), pp. 18–22. doi: 10.1108/10650751111106528.

Chen, W. C. W. *et al.* (2016) 'HHS Public Access', 33(2), pp. 557–573. doi: 10.1002/stem.1868.Human.
Choetkiertikul, M. *et al.* (2018) 'A deep learning model for estimating story points', *IEEE Transactions on Software Engineering*. doi: 10.1109/TSE.2018.2792473.

Chulani, S., Boehm, B. and Steece, B. (1999) 'Bayesian analysis of empirical software engineering cost models', *IEEE Transactions on Software Engineering*, 25(4), pp. 573–583. doi: 10.1109/32.799958.

Dhingra, B. *et al.* (2017) 'A Comparative Study of Word Embeddings for Reading Comprehension', (3). doi: 10.3170/2007-8-18431.

Gers, F. A., Schmidhuber, J. and Cummins, F. (2000) 'Learning to forget: Continual prediction with LSTM', *Neural Computation*, 12(10), pp. 2451–2471. doi: 10.1162/089976600300015015.

Graves, A., Mohamed, A. R. and Hinton, G. (2013) 'Speech recognition with deep recurrent neural networks', *ICASSP, IEEE International Conference on Acoustics, Speech and Signal Processing - Proceedings*. IEEE, (6), pp. 6645–6649. doi: 10.1109/ICASSP.2013.6638947.

Hearty, P. *et al.* (2009) 'Predicting project velocity in XP using a learning dynamic Bayesian network model', *IEEE Transactions on Software Engineering*. IEEE, 35(1), pp. 124–137. doi: 10.1109/TSE.2008.76.

Heravi, A., Coffey, V. and Trigunarsyah, B. (2015) 'Evaluating the level of stakeholder involvement during the project planning processes of building projects', *International Journal of Project Management*. Elsevier Ltd, 33(5), pp. 985–997. doi: 10.1016/j.ijproman.2014.12.007.

Hochreiter, S. and Frasconi, P. (2009) 'Gradient Flow in Recurrent Nets: The Difficulty of Learning LongTerm Dependencies', *Learning*, pp. 401–403. doi: 10.1109/9780470544037.ch14.

Mahnic, V. (2011) 'A case study on agile estimating and planning using scrum', *Elektronika ir Elektrotechnika*, (5), pp. 123–128. doi: 10.5755/j01.eee.111.5.372.

Mikolov, T. *et al.* (2017) 'Advances in Pre-Training Distributed Word Representations', (1). doi: 10.1589/jpts.28.2114.

Moharreri, K. *et al.* (2016) ‘Cost-Effective Supervised Learning Models for Software Effort Estimation in Agile Environments’, *Proceedings - International Computer Software and Applications Conference*, 2, pp. 135–140. doi: 10.1109/COMPSAC.2016.85.

Olmer, P. (2001) ‘Knowledge Discovery : Data and Text Mining’, pp. 47–55.

Panda, A., Satapathy, S. M. and Rath, S. K. (2015) ‘Neural Network Models for Agile Software Effort Estimation based on Stor Points’, *Proceeding of the International Conference on Advances in Computing, Control and Networking*, 57(57), pp. 772–781. doi: 10.15224/978-1-63248-038-5-06.

Pascanu, R., Mikolov, T. and Bengio, Y. (2013) ‘On the difficulty of training recurrent neural networks’, *Phylogenetic Diversity*, (2), pp. 41–71. doi: 10.1109/72.279181.

Porru, S. *et al.* (2016) ‘Estimating Story Points from Issue Reports’, *Proceedings of the The 12th International Conference on Predictive Models and Data Analytics in Software Engineering - PROMISE 2016*, pp. 1–10. doi: 10.1145/2972958.2972959.

Qi, Y. *et al.* (2018) ‘When and Why are Pre-trained Word Embeddings Useful for Neural Machine Translation?’, pp. 529–535. doi: 10.1002/adma.201200431.

Risch, J. and Krestel, R. (2018) ‘Aggression Identification Using Deep Learning and Data Augmentation’, *Proceedings of the First Workshop on Trolling, Aggression and Cyberbullying (TRAC-2018)*, pp. 77–85. Available at: http://aclweb.org/anthology/W18-44%0A900.

Scheepers, T., Kanoulas, E. and Gavves, E. (2018) ‘Improving Word Embedding Compositionality using Lexicographic Definitions’, *Proceedings of the 2018 World Wide Web Conference on World Wide Web - WWW ’18*, pp. 1083–1093. doi: 10.1145/3178876.3186007.

Usman, M. *et al.* (2014) ‘Postprint Effort Estimation in Agile Software Development : A Systematic Literature Review’, pp. 82–91. doi: 10.1145/2639490.2639503.

Wang, Y. *et al.* (2018) ‘A comparison of word embeddings for the biomedical natural language processing’, *Journal of Biomedical Informatics*, 87, pp. 12–20. doi: 10.1016/j.jbi.2018.09.008.

Wieting, J. *et al.* (2015) ‘From Paraphrase Database to Compositional Paraphrase Model and Back’. doi: 10.1016/S0022-3115(04)00526-4.

Xiong, C., Merity, S. and Socher, R. (2016) ‘Dynamic Memory Networks for Visual and Textual Question Answering’, 48(2015). doi: 10.1007/s00158-007-0225-0.

Zhou, P. *et al.* (2016) ‘Text Classification Improved by Integrating Bidirectional LSTM with Two-dimensional Max Pooling’, 2(1). Available at: http://arxiv.org/abs/1611.06639.