Decentralized Large-Scale Natural Language Processing Using Gossip Learning

Alkathiri, Abdul Aziz

Award date:
2020

Awarding institution:
Royal Institute of Technology

DEGREE PROJECT IN COMPUTER SCIENCE AND ENGINEERING, SECOND CYCLE, 30 CREDITS

STOCKHOLM, SWEDEN 2020

Decentralized Large-Scale Natural Language Processing Using Gossip Learning

ABDUL AZIZ ALKATHIRI

KTH ROYAL INSTITUTE OF TECHNOLOGY
SCHOOL OF ELECTRICAL ENGINEERING AND COMPUTER SCIENCE

Decentralized Large-Scale Natural Language Processing Using Gossip Learning

ABDUL AZIZ ALKATHIRI

Master's Programme, Computer Science, 120 credits
Date: August 21, 2020
Supervisor: Magnus Sahlgren (RISE SICS) Lodovico Giaretta (KTH)
Examiner: Sarunas Girdzijauskas
School of Electrical Engineering and Computer Science
Host company: RISE SICS
Swedish title: Decentraliserad Storskalig Naturlig Språkbehandling med Hjälp av Skvallerinlärning

Decentralized Large-Scale Natural Language Processing Using Gossip Learning / Decentraliserad Storskalig Naturlig Språkbehandling med Hjälp av Skvallerinlärning

Abstract

The field of Natural Language Processing in machine learning has seen rising popularity and use in recent years. The nature of Natural Language Processing, which deals with natural human language and computers, has led to the research and development of many algorithms that produce word embeddings. One of the most widely-used of these algorithms is Word2Vec. With the abundance of data generated by users and organizations and the complexity of machine learning and deep learning models, performing training using a single machine becomes unfeasible. The advancement in distributed machine learning offers a solution to this problem. Unfortunately, due to reasons concerning data privacy and regulations, in some real-life scenarios, the data must not leave its local machine. This limitation has lead to the development of techniques and protocols that are massively-parallel and data-private. The most popular of these protocols is federated learning. However, due to its centralized nature, it still poses some security and robustness risks. Consequently, this led to the development of massively-parallel, data private, decentralized approaches, such as gossip learning. In the gossip learning protocol, every once in a while each node in the network randomly chooses a peer for information exchange, which eliminates the need for a central node. This research intends to test the viability of gossip learning for large-scale, real-world applications. In particular, it focuses on implementation and evaluation for a Natural Language Processing application using gossip learning. The results show that application of Word2Vec in a gossip learning framework is viable and yields comparable results to its non-distributed, centralized counterpart for various scenarios, with an average loss on quality of 6.904%.

Keywords

gossip learning, decentralized machine learning, distributed machine learning, NLP, Word2Vec, data privacy

Sammanfattning

Fältet Naturlig Språkbehandling (Natural Language Processing eller NLP) i maskininlärning har sett en ökande popularitet och användning under de senaste åren. Naturen av Naturlig Språkbehandling, som bearbetar naturliga mänskliga språk och datorer, har lett till forskningen och utvecklingen av många algoritmer som producerar inbäddningar av ord. En av de mest använda av dessa algoritmer är Word2Vec. Med överflödet av data som genereras av användare och organisationer, komplexiteten av maskininlärning och djupa inlärningsmodeller, blir det omöjligt att utföra utbildning med hjälp av en enda maskin. Avancemangen inom distribuerad maskininlärning erbjuder en lösning på detta problem, men tyvärr får data av sekretesskäl och datareglering i vissa verkliga scenarier inte lämna sin lokala maskin. Denna begränsning har lett till utvecklingen av tekniker och protokoll som är massivt parallella och dataprivata. Det mest populära av dessa protokoll är federerad inlärning (federated learning), men på grund av sin centraliserade natur utgör det ändock vissa säkerhets- och robusthetsrisker. Följaktligen ledde detta till utvecklingen av massivt parallella, dataprivata och decentraliserade tillvägagångssätt, såsom skvallerinlärning (gossip learning). I skvallerinlärningsprotokollet väljer varje nod i nätverket slumpmässigt en like för informationsutbyte, vilket eliminerar behovet av en central nod. Syftet med denna forskning är att testa livskraftigheten av skvallerinlärning i större omfattningens verkliga applikationer. I synnerhet fokuserar forskningen på implementering och utvärdering av en NLP-applikation genom användning av skvallerinlärning. Resultaten visar att tillämpningen av Word2Vec i en skvallerinlärnings ramverk är livskraftig och ger jämförbara resultat med dess icke-distribuerade, centraliserade motsvarighet för olika scenarier, med en genomsnittlig kvalitetsförlust av 6,904%.

Nyckelord

skvallerinlärning, decentraliserad maskininlärning, distribuerad maskininlärning, naturlig språkbehandling, Word2Vec, dataintegritet

Acknowledgments

First and foremost, I would like to thank my supervisor at KTH, Lodovico Giaretta, for his supervision, guidance, and support; his balanced approach of giving me autonomy and being critical when necessary is a vital contributing factor in achieving the initial goals and completion of this work.

Furthermore, I would like to give my gratitude and appreciation to my examiner, Šarūnas Girdzijauskas, whose constructive criticisms positively made me push the limits of the scope and quality of this thesis, as well as my supervisor at RISE, Magnus Sahlgren, who steered us in the right direction and provided us with much-needed industry knowledge. Last but certainly not least, I would like to thank Ahmed Emad Samy Yossef Ahmed for his insights into the field of NLP.

Stockholm, August 2020
Abdul Aziz Alkathiri

Contents

1 Introduction **1**
1.1 Background . . . 1
1.2 Problem . . . 3
1.2.1 Privacy Concerns . . . 3
1.2.2 Robustness of Centralized Approaches . . . 3
1.2.3 Trade-off between Information Sharing and Quality . . . 3
1.3 Purpose . . . 4
1.4 Goals . . . 4
1.5 Research Methodology . . . 5
1.6 Delimitations . . . 6
1.7 Overall Results . . . 6
1.8 Structure of the Thesis . . . 7

2 Related Work **9**
2.1 Distributed and Decentralized Machine Learning . . . 9
2.1.1 Distributed Machine Learning . . . 9
2.1.2 Federated Learning . . . 11
2.1.3 Decentralized Machine Learning . . . 12
2.1.4 Gossip Learning . . . 13
2.2 Word2Vec . . . 15

3 Methodology **17**
3.1 Algorithms . . . 17
3.1.1 Word2Vec: Skip-gram . . . 17
3.1.2 Softmax Function . . . 19
3.1.3 Noise Contrastive Estimation . . . 20
3.1.4 Gossip Learning . . . 20
3.2 Research Paradigm . . . 22
3.3 Data Collection . . . 23

3.3.1 Sampling 23
3.4 Experimental Setup 24
3.4.1 Hardware and Software Used 24
3.4.2 Libraries Used 25
3.5 Evaluation framework 25

4 Experiments **27**
4.1 Simulation Model Parameters 27
4.2 Setup and Implementation 29
4.2.1 Traditional Centralized Training 29
4.2.2 Gossip Learning with Frequent Exchange 29
4.2.3 Gossip Learning with Infrequent Exchange 30
4.2.4 Local Nodes Training with Common Vocabulary . . . 31

5 Results and Analysis **33**
5.1 Traditional Centralized Training 33
5.2 Gossip Learning with Frequent Exchange 34
5.3 Gossip Learning with Infrequent Exchange 46
5.4 Local Nodes Training with Common Vocabulary 59
5.5 Discussion 66

6 Conclusions and Future work **67**
6.1 Conclusions 67
6.2 Limitations 69
6.3 Future work 69
6.4 Reflections 70

References **71**

A List of stop words **75**

List of Figures

2.1 In the data parallelism approach, multiple instances of the same model are trained on different parts of the dataset, while in model parallelism the model is distributed along the nodes instead. 10
2.2 Word2Vec is a two-layer neural network. 16

3.1 The Skip-gram architecture. Skip-gram's training objective is to learn representations that predict nearby words well. 18
3.2 The creation of training pairs for the Skip-gram model. 19

4.1 Learning rate history. 28

5.1 Traditional centralized learning results: $loss$ and $w2v_{sim}$ values over batches. The $w2v_{sim}$ value converges to 65. 34
5.2 Gossip learning with frequent exchange (*topicwise*) results: $loss$ values over local batches. 36
5.3 Gossip learning with frequent exchange (*topicwise*) results: $w2v_{sim}$ values over local batches. The $w2v_{sim}$ value converges to 60. In comparison, the baseline is 65. 37
5.4 Gossip learning with frequent exchange (*topicwise*) model sharing. 37
5.5 Gossip learning with frequent exchange (*randombalanced*) results: $loss$ values over local batches. 39
5.6 Gossip learning with frequent exchange (*randombalanced*) results: $w2v_{sim}$ values over local batches. The $w2v_{sim}$ value converges to 60. In comparison, the baseline is 65. 40
5.7 Gossip learning with frequent exchange (*randomimbalanced*) results: $loss$ values over local batches. 42

5.8 Gossip learning with frequent exchange (*randomimbalanced*) results: $w2v_{sim}$ values over local batches. The $w2v_{sim}$ value converges to 60. In comparison, the baseline is 65. 42
5.9 Gossip learning with frequent exchange (*half*) results: $loss$ values over local batches. 43
5.10 Gossip learning with frequent exchange (*half*) results: $w2v_{sim}$ values over local batches. The final $w2v_{sim}$ values range between 22 and 31. In comparison, the baseline is 65. 45
5.11 Gossip learning with frequent exchange (*half*) model sharing. 45
5.12 Gossip learning with infrequent exchange (*topicwise*) results: $loss$ values over local batches. 48
5.13 Gossip learning with infrequent exchange (*topicwise*) results: $w2v_{sim}$ values over local batches. The $w2v_{sim}$ value converges to 62. In comparison, the baseline is 65. 49
5.14 Gossip learning with infrequent exchange (*topicwise*) model sharing. 49
5.15 Gossip learning with infrequent exchange (*randombalanced*) results: $loss$ values over local batches. 50
5.16 Gossip learning with infrequent exchange (*randombalanced*) results: $w2v_{sim}$ values over local batches. The $w2v_{sim}$ value converges to 61. In comparison, the baseline is 65. 52
5.17 Gossip learning with infrequent exchange (*randomimbalanced*) results: $loss$ values over local batches. 52
5.18 Gossip learning with infrequent exchange (*randomimbalanced*) results: $w2v_{sim}$ values over local batches. The $w2v_{sim}$ value converges to 61. In comparison, the baseline is 65. 54
5.19 Gossip learning with infrequent exchange (*half*) results: $loss$ values over local batches. 56
5.20 Gossip learning with infrequent exchange (*half*) results: $w2v_{sim}$ values over local batches. The final $w2v_{sim}$ values range between 17 and 35. In comparison, the baseline is 65. 57
5.21 Local nodes training with common vocabulary results: $loss$ values over local batches. 59
5.22 Local nodes training with common vocabulary results: $w2v_{sim}$ values over local batches. The final $w2v_{sim}$ values range between 39 and 57. In comparison, the baseline is 65. 60

List of Tables

5.1 Target words and predicted context words for traditional centralized training at batch 5,000,000. 35
5.2 Target words and predicted context words for gossip learning with frequent exchange *topicwise* at each local batch 500,000. . 38
5.3 Target words and predicted context words for gossip learning with frequent exchange *randombalanced* at each local batch 500,000. 41
5.4 Target words and predicted context words for gossip learning with frequent exchange *randomimbalanced* at each local batch 500,000. 44
5.6 Summary of sub-configuration results of gossip learning with frequent exchange. 46
5.5 Target words and predicted context words for gossip learning with frequent exchange *half* at each local batch 500,000. . . . 47
5.7 Target words and predicted context words for gossip learning with infrequent exchange *topicwise* at each local batch 500,000. 51
5.8 Target words and predicted context words for gossip learning with infrequent exchange *randombalanced* at each local batch 500,000. 53
5.9 Target words and predicted context words for gossip learning with infrequent exchange *randomimbalanced* at each local batch 500,000. 55
5.10 Target words and predicted context words for gossip learning with infrequent exchange *half* at each local batch 500,000. . . 58
5.11 Summary of sub-configuration results of gossip learning with infrequent exchange. 59
5.12 Target words and predicted context words for node 2 *science* for local nodes training with common vocabulary at batch 500,000. 61

5.13 Target words and predicted context words for node 4 *politics* for local nodes training with common vocabulary at batch 500,000. 62
5.14 Target words and predicted context words for node 5 *business* for local nodes training with common vocabulary at batch 500,000. 63
5.15 Target words and predicted context words for node 8 *humanities* for local nodes training with common vocabulary at batch 500,000. 64
5.16 Target words and predicted context words for node 9 *history* for local nodes training with common vocabulary at batch 500,000. 65

List of acronyms and abbreviations

CBOW Continuous Bag of Words Model

GDPR General Data Protection Regulation

NCE Negative Contrastive Estimation

NLP Natural Language Processing

NLTK Natural Language Toolkit

Chapter 1

Introduction

1.1 Background

The growth of the study and use of machine learning in recent years in both academia and industry can be attributed, among other things, to the abundance of data available. The capabilities of machine learning algorithms to learn and understand models that represent complex systems have advanced exponentially.

In spite of these advances, machine learning models, and especially deep learning models [1], are used to represent complex systems, they require huge amounts of data. Because a single machine is often not enough in terms of storage and computational power for these models, the need for developing scalable, distributed machine learning approaches that executes training in parallel [2] as opposed to traditional centralized solutions where the data fits in a single machine, has arisen.

Typically, the scenario of where data is moved to a datacenter to perform distributed training, is utilized to overcome the limitations of a single machine. Unfortunately, there are some drawbacks to the use of the datacenter-scale approach, such as moving the data from local machines and storing it in the data center, which is costly and sometimes logistically tedious. More importantly, when dealing with sensitive data, collecting data in a central location increases the risks due to concerns for data privacy. Therefore, this topic is becoming more relevant and important as a large number of users and organizations are concerned about data privacy and more regulations on this matter being drafted. Therefore, for the sake of scalability and privacy, there is a call for massively-parallel, data-private approaches - or distributed approaches where training is done directly on the machines that produce and

hold the data, without having to share or transfer it.

The main approach for such massively-parallel, data-driven techniques is federated learning [3], a centralized approach where a central server coordinates the activity of the nodes in the network, all while the local data are not shared between the nodes.

Massively-parallel, data-private strategies using centralized approaches however, such as federated learning, are plagued with issues such as the presence of a central node which may act as a privileged "gatekeeper", as well as reliability issue on the account of that central node.

Federated learning, while it offers a distributed, scalable and private machine learning environment, still poses some concerns with regards to robustness and privacy due its centralized nature. It is therefore interesting for researchers to look into decentralized approaches that are scalable, robust and privacy-preserving for large-scale real-world applications.

The limitations and apparent shortcomings of massively-parallel, data-private centralized approaches have given rise to decentralized approaches, where no single node acts as the central gatekeeper, such as gossip learning [4]. The basic idea behind gossip learning is it is a data-private technique that requires the nodes to share their models with each other once in a while.

This therefore creates a juxtaposition between the approach of the datacenter-scale scenario versus massively-parallel, data-private scenario. Within the context of this project however, distributed machine learning shall refer to the latter and any reference to centralized or decentralized approaches or techniques (although they are applicable in both of scenarios) shall be in the context of the latter as well unless stated otherwise. The juxtaposition therefore here is whether the data is kept at their local machines and devices or otherwise; in other words, periodic, lightweight communication while data the data always stays at the local machines.

To the best of our knowledge one area of machine learning that has been unexplored in the context of massively-parallel, data-private scenario, is Natural Language Processing (NLP), which is the study of the interactions between computers and natural(human) languages [5].

One of the most widely-used of NLP algorithms is Word2Vec [6]. The basic premise of the Word2Vec approach builds upon the assumption that words that appear frequently together are similar. Word2Vec is a two-layer neural network which groups vectors of similar words in a vector space. More details of Word2Vec are given in Section 2.2.

1.2 Problem

1.2.1 Privacy Concerns

While the privacy, robustness and quality concerns are shared by all potential applications of distributed machine learning, the focus in this project will be on one particular application in the NLP. That is, the case where a small number of separate organizations (such as, for example, government agencies) want to train a powerful NLP model, using the combined data of their corpora, but without sharing them, as that could potentially violate privacy laws or data collection agreements. And in the case where these organizations are private companies in lieu of government agencies, an additional concern presents itself in the form of the leak of strategic information to other companies resulting from this cooperation.

These organizations wish to benefit from each other's wealth of corpora while minimizing the risk of disclosing the private contents of those corpora. Traditional centralized machine learning configurations necessitate that the corpora be brought to a central node where training will take place. Likewise with the datacenter-scale approach, the data must be brought to a datacenter.

And while massively-parallel, data-private, decentralized approaches like gossip learning do not guarantee the preservation of the privacy of the contents, they minimize the chances of the inferring the contents from metadata.

1.2.2 Robustness of Centralized Approaches

Another problem that may arise from using centralized approaches is robustness. Centralized methods such as federated learning are dependent on the availability of the central node. Decentralized approaches, on the other hand, handle node failures better.

It is therefore interesting to query the whether the quality loss that gossip learning may incur (if significantly or any at all) is worth the probable downtime of comparable traditional centralized methods.

1.2.3 Trade-off between Information Sharing and Quality

Since gossip learning approach necessitates the exchange of models between nodes once in a while, for large models, this can be bandwidth-intensive. How often this exchange happens is of concern. While more frequent exchanges may result in faster convergence of the models, the bandwidth used to do so

will bear the brunt of the cost. Thus, it is interesting to investigate how much the quality of the trained models is affected by reducing the frequency of the sharing of the models.

Therefore, the main question of this work is

> ***How do models that are produced from the corpus of each node on a decentralized, fully-distributed, data-private configuration, i.e. gossip learning, compare to that trained using a traditional centralized approach where all the data are moved from the local machines or devices using comparable parameters with respect to several evaluations?***

1.3 Purpose

The purpose of this project is to test the viability and gauge the performance of a real-world application of an NLP algorithm (Word2Vec [6] in particular), running on a decentralized, fully-distributed configuration, data-private approach, i.e. gossip learning, with respect to a counterpart centralized setting. In particular, an application where privacy preservation is to be taken into account. This is driven by the need of organizations to make use of the corpora from other organizations for the purpose of NLP training, all without disclosing their own contents that are deemed sensitive. Moreover, the gossip learning configuration is further compared under different circumstances pertaining to node size and topicality, frequency of of model sharing, and how much the models share with each other.

However, in order to achieve the purpose and goals of this project, several assumptions are made, some of which are in line with the assumptions given by the gossip learning approach [4]. The assumptions are further detailed in Section 3.1.4.

1.4 Goals

Pursuant to the stated purpose, the project is carried out guided by the following goals and deliverables:

1. implementation of Word2Vec algorithm using the gossip learning approach and evaluation of its viability for large-scale applications;

2. evaluation and comparison between the performance of said algorithm on gossip learning to its traditional centralized counterpart;

3. execution of tests of said algorithm on gossip learning under different circumstances of parameters and datasets as introduced in 1.3 and their comparison.

1.5 Research Methodology

In order to fulfill its purpose, this project needs to implement the NLP algorithm, i.e. Word2Vec, on a dataset of real-world scale under different configurations of parameters, where these configurations are bound by the assumptions made and are tuned to achieve the purpose and goals of this project. Because of the novelty of the research area - to the best of our knowledge - this project aims to shed light on the performance and cost trade-off between running the NLP algorithm in a traditional centralized fashion and on a gossip learning configuration. Therefore, the baseline for evaluation is the word embedding model in a centralized setting.

The evaluation is to be done on the gossip learning approach itself as well as the trained embeddings; how much longer it takes to train models in the gossip learning configuration comparatively, the costs in terms of bandwidth for data transfer and the effect of its frequency on models' characteristics, and the general viability of using gossip learning to train sensitive corpora are all interesting questions with respect to the approach itself.

Furthermore, evaluating the embedding models is the other dimension of the overall evaluation in this project. In addition to the training process to be evaluated, as explained in the previous paragraph, the quality of the produced embeddings will also be a subject of evaluation by comparing them with those to be obtained in a traditional centralized configuration using the same dataset and hyperparameters.

Therefore, another evaluation method that is used to compare the centralized model as well as the models trained using gossip learning is using pre-trained models of the same embedding dimensions. This allows for the comparison of the models with respect to an external reference.

The choice of examining a relatively outdated NLP technique that is Word2Vec [6] is because - to the best of our knowledge - investigation into applying gossip learning on NLP techniques have not been thoroughly explored. It is therefore appropriate to start with one of the most popular and attested method, which is easier to understand and interpret since people have

gathered more experience on it.

Further details of the research methodology and approaches, as well as the datasets used are presented in Chapter 3.

1.6 Delimitations

Due to the scope and the novelty of the research area, the focus of this project is to find out whether the NLP algorithm can learn high-quality embeddings at reasonable speeds in a decentralized, data-private setting based on gossip learning. Therefore, despite preliminary exploration and experimentation, exploration into all possible scenarios is limited and will be left to potential future works that build upon the findings and results of this project, which will be explained further in Section 6.3.

This project aims to explore the viability of running an NLP algorithm on gossip learning and compare it to its centralized counterpart. For this reason, all possible potential scenarios that may arise are not explored. For instance, the network used for communication is assumed perfect and that nodes will not drop due to network issues. Further, asynchronous communication rounds between the nodes will not be investigated in this project. So, possible network issues will not be examined.

This project does not aim to compare various NLP tasks but rather it aims to focus on one using the different approaches. The intuition behind it is that the results and findings may be extended to other tasks as well as other machine learning applications, and this enables us to focus more on that particular task.

Furthermore, this project is not geared towards finding the most optimal hyperparameters of the algorithms used, albeit it would be an area of interest for future research, nor does it focus on certain aspects of gossip learning, such as asynchronicity and network connectivity, as these are orthogonal problems and are covered by other literature.

1.7 Overall Results

Overall, the results of this project show that the quality of word embedding produced using the gossip learning approach is comparable to that trained using a traditional, centralized approach, even when the frequency of communications has been reduced by a factor of 50 - from 50,000 rounds of communication between the nodes to 1,000; more specifically, with an average loss on quality

of 6.904%. This confirms the viability of using the gossip learning approach for large-scale, real-world, NLP applications.

1.8 Structure of the Thesis

This chapter introduced the topic, motivation, methodology, and limitations of this project. Chapter 2 further expands on the background and describes the related studies. Chapter 3 details the methods used in this project and dataset selection and preparation, while Chapter 4 gives details on the experiments implementation. Chapter 5 shows the results and analysis of the experiments and discusses the findings of this project. Finally, Chapter 6 suggests potential future directions of research.

Chapter 2

Related Work

2.1 Distributed and Decentralized Machine Learning

2.1.1 Distributed Machine Learning

In addition to the availability of large datasets such as ImageNet [7] and Open Images [8] for training relatively complex models, what has lead to the advances in the field of machine learning and deep learning is the increase in available computational power to train these models; in particular, the advances made in GPUs as the source to perform computation [9] and optimization of parallel computation on GPUs [10].

Owing to these advances in computational resources as well as optimization in algorithms, a GPU is a powerful tool to train machine learning models. However, the complexity of some deep learning models and the size of data required to train them have called for distributed training.

This naturally means using multiple GPUs or machines to carry out training, also known as datacenter-scale approach. However, distributed training setups, such as one in a datacenter, mean that the data has to be transferred from the local machines or devices where they are originally from, and this is a breach of privacy in some cases where the data must not leave these machines. It is worth mentioning however, that using a single GPU on a remote server where data has to be transferred from the local machines poses the same privacy issues.

In general, for distributed-scale machine learning, there are three orthogonal categorizations of distributed machine learning: in terms of its *parallelism*, *synchronicity*, and *topology*. Researchers have categorized distributed machine

learning frameworks with regards to its type of distribution or *parallelism*, namely data parallelism and model parallelism [11]. In the data parallel approach, the data is partitioned into as many parts as there are nodes in the system and all nodes then apply the same algorithm to the different parts of the dataset. This approach is more commonly used as it is likely that models fit on a single node while the datasets do not. However, when the model size for instance, does not fit in a single node memory, model parallelism can be used instead. In the model parallel approach, each needs the entire dataset. The model is then just the aggregate of the parts in the various nodes. These two approaches are by no means mutually exclusive [2]. Figure 2.1 illustrates the two approaches.

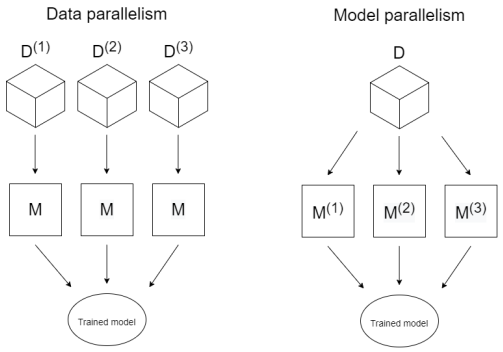

Figure 2.1: In the data parallelism approach, multiple instances of the same model are trained on different parts of the dataset, while in model parallelism the model is distributed along the nodes instead.

Another way to categorize distributed machine learning is by its *synchronicity*. In theory, in distributed configuration, computational steps of the training are performed in parallel across the nodes, and immediate communication between the nodes is not necessary after each local step. These communication rounds however, can be handled by two approaches: synchronous or asynchronous [12].

Using the synchronous approach, all the nodes start executing out a number of iterations of computation. Then each node waits for all the other nodes to complete their execution before all the nodes then share and aggregate the results. Then, this is repeated. Using this approach guarantees that the all the nodes will have received the required results produced during the last set of iterations before moving on, therefore guaranteeing quicker convergence, i.e. within fewer iterations thus less required time. The downside however, is that

this approach is vulnerable to node failures, and late nodes (or stragglers) can hold up the training process.

On the other hand, in the asynchronous approach, each nodes computes an update and shares the results back to the network as soon as possible. This gives node independence from each other. This very simply solves the issue with node failure and straggler nodes. On the other hand however, the update sent by some nodes could have been computed based on a different (older) model and thus it takes the other partial models or global model away from convergence. This problem is fortunately mitigated by higher number of nodes in the network.

Additionally, distributed machine learning can be categorized based on its *topology*, or how the nodes within the network are organized. For instance, in *centralized systems*, the aggregation happens in a single central node. *Decentralized systems* such as those with tree-like topologies which allow for the intermediate aggregation, where each node communicates only with its parent and child nodes. And on the other end of the spectrum, in *fully decentralized systems*, each node has its own copy of the parameters and the nodes communicate with each other directly.

Practically, these categorizations overlap with each other and their combination affects the amount of communication required for training.

However, with the distributed machine learning framework, in particular using the datacenter-scale approach, there is still the issue of data privacy since this scenario requires the moving of data which is deemed sensitive from local machines to servers. Hence, the need for a data-private approach. The most widely-used technique for massively-parallel, data-private machine learning is federated learning.

2.1.2 Federated Learning

The increasing need for the privacy preservation of data can be attributed to many different factors. First, the value of data collected of online users has increased significantly, as data is a major commodity used to predict user behavior and inform business decisions. Also, users have put a bigger emphasis on data privacy due to the major scandals involving the use of their data. Finally, legislation such as the General Data Protection Regulation (GDPR) [13] obligates parties that collect data to inform and get the consent of users with regards to their data's privacy and use.

In line with privacy preservation of data, distributed approaches have been gaining more attention by both research and industry. One of these approaches

is federated learning first introduced by McMahan et al. [14]. This approach allows the training of a global model based on computations of node devices in the network without disclosure of data from the nodes. Federated learning is a centralized, data-parallel, synchronous approach.

Algorithm 2.1 shows the generic algorithm for both the central node (computational node) and the worker nodes (data nodes). In each iteration, the central node sends the current global model out to the worker nodes. Each worker node then can calculate an update of a the model based on the local data. This update is then sent back to the central server which aggregates all these updates to produce an updated global model.

Algorithm 2.1: Generic Federated Learning Algorithm

```
procedure ServerLoop
    loop
        S ← RandomSubset(devices, K)
        forall k ∈ S do
            SendToDevice(k, currentModel)
        end
        forall k ∈ S do
            w_k, n_k ← ReceiveFromDevice(k)
        end
    end loop
end procedure
procedure OnModelReceivedByDevice(model)
    model ← model + Update(model, localData)
    SendToServer(model, SizeOf(localData))
end procedure
```

However, despite the data privacy characteristics of federated learning, it is centralized approach, which, as will be detailed in the next section, can still pose some privacy concerns, such as robustness and the presence of a gatekeeping central node. Decentralized approaches can mitigate these problems.

2.1.3 Decentralized Machine Learning

With decentralized machine learning, there is no central gatekeeper that has full control over the whole process of training. Therefore, all the nodes in the network execute the same protocols with the same level of privileges. This mitigates the chances of exploitation by malicious actors. Thus, decentralized machine learning is characterized by transparency and independence.

It has been shown that malicious attackers can extract coherent data from training datasets from the trained models and that measures of anonymization can be effectively undone [15] [16]. Therefore, there is a huge interest in looking for ways that bolster the privacy-preserving capability of training model without significant impact in the models quality. With respect to the privacy of the data used for training, the characteristic of decentralized machine learning can be useful to provide better privacy preservation compared to centralized approach. And it is not hard to imagine a malicious gatekeeper node or an entity attacking that central node in a centralized machine learning configuration can use its privileges as the gatekeeper to extract sensitive data.

Decentralized machine learning also scales better comparatively and is more flexible. A *distributed* machine learning protocol would face scalability issues at a certain network size. With a peer-to-peer network protocol, decentralized machine learning can virtually scale up to unlimited sizes and be more fault-tolerant.

These characteristics make decentralized machine learning protocols that allow a network of nodes train a machine learning model with partial datasets without exchanging the contents worth investigating further. Unfortunately, exploration into this topic has not been given a lot of attention, with only a few algorithms [4] [17] [18] and pushing the limits of such protocols [19] have been published.

There are two general strategies when it comes to the optimization of decentralized machine learning. The first is decentralized averaging [20], where the approach to solving the problem is done through training models locally and averaging them throughout the network, such as the gossip learning protocol by Ormándi et al. [4]. The second strategy is decentralized optimization [21], where a single model is cooperatively built by taking the sum of the losses locally for each node and minimizing the global loss.

2.1.4 Gossip Learning

The gossip communication approach refers to a set of decentralized communication protocols inspired by the behaviour of the spread of gossip socially among people [22]. First introduced for the purpose of efficiently synchronizing distributed servers [23], it has also been applied to different problems, such as data aggregation [20], and failure detection [24].

Algorithm 2.2 shows the generic gossip-based protocol, where as per the core principle of gossip learning, every once in a while, each node in a network randomly chooses a peer for information exchange. The implementation of

ExchangeInformation depends of the purpose on the protocol.

Algorithm 2.2: Generic Gossip-based Protocol

```
loop
    Wait(Δ)
    p ← ChooseRandomPeer()
    ExchangeInformation(p)
end loop
```

Gossip learning, introduced by Ormándi et al. [4], employs the gossip protocol based on random walks. In contrast to federated learning, it is an asynchronous, data-parallel, decentralized averaging approach. Gossip learning has been shown to be effective when applied to various machine learning techniques, including binary classification with support vector machines [4], k-means clustering [25], and low-rank matrix decomposition [26]. However, these implementations of gossip learning have been limited to the scenario where each node in the network only holds a single-data point.

Algorithm 2.3 shows the generic algorithm of gossip learning. As long as the loop runs, it gossips the current model to a randomly chosen peer. The procedure runs passively, that is, upon receiving a model. CreateModel creates an updated model based on the received model and the model received last - as each node saves the last model received as shown in Algorithm 2.2; this can be done by averaging both models for instance, but the details of how the models are aggregated internally depends on the specific problem and method employed.

Algorithm 2.3: Generic Gossip Learning Algorithm

```
currentModel ← InitModel()
lastModel ← currentModel
loop
    Wait (Δ)
    p ← RandomPeer()
    SEND(p, currentModel)
end loop
procedure OnModelReceivedByDevice(m)
    currentModel ← model + CreateModel(m, lastModel)
    lastModel ← m
end procedure
```

Algorithm 2.4 shows three possible implementation of CreateModel. Update creates a new updated model based on local data, while Merge joins

two models into one, either by averaging or otherwise. The most naive of these implementations is CreateModelRW, as no merging of models is taking place. On the other hand, CreateModelUM and CreateModelMU perform update and merging of models but in different orders. According to Ormándi et al. [4], the latter gives a better performance as each model is updated on a different node; this maintains node independence.

Algorithm 2.4: Implementations of CreateModel

```
function CreateModelRW(m_1, m_2)
    return Update(m_1)
end function
function CreateModelUM(m_1, m_2)
    return Merge(Update(m_1), Update(m_2))
end function
function CreateModelMU(m_1, m_2)
    return Update(Merge(m_1, m_2)
end function
```

The implementations of gossip learning so far have been limited among other things in that the scenarios considered are impractical for applications in industrial scale. For instance, Giaretta [19] showed that the gossip protocol fails to give favorable results when exposed to certain conditions that appear in some real-world scenarios, such as bias towards the data stored with faster communication speeds and the impact of topologies on the convergence speed of models.

It is worth noting however, that distributed approaches - whether centralized or decentralized - that do not necessitate the move of data from the local nodes prevents obvious and easy exploits. However, it is not a panacea; determined attackers can still infer some features of data content regardless of using this approach.

2.2 Word2Vec

Word embedding is the process of embedding words into a vector space. Each word is thus associated with a vector in such a way that the similarities between words are reflected through the similarities between vectors. Therefore, these vectors are called *word embeddings* or *word vectors*.

NLP is the study of the interactions between computers and natural (human) languages [5]. Research within this area includes speech recognition,

natural language understanding, and natural language generation. In the context of NLP tasks, word embedding has seen advances in the application of machine learning tools such as neural networks to language related tasks [27] [28] [29].

Introduced by Mikolov et al. [30], Word2Vec refers to a group of models that produce word embeddings, which are used for learning vector representation of words. The Word2Vec approach builds on the assumption that words that frequently appear in similar contexts have similar syntactical and semantic roles. As the name suggests, Word2Vec groups vectors of similar words together in a vector space. Word2Vec is a two-layer neural network (as shown in Figure 2.2), where the input is a one-hot encoded word and the output is a weight matrix of vectors, each vector representing a word in the vocabulary.

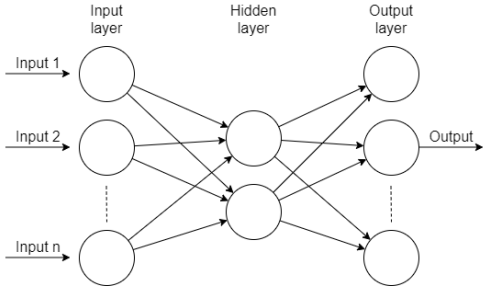

Figure 2.2: Word2Vec is a two-layer neural network.

Mikolov et al. [30] proposed two model architectures: Continuous Bag of Words Model (CBOW) Model and Continuous Skip-gram Model. While CBOW takes the context of each word as the input and predicts that word, on the contrary, Skip-gram is formulated as a classification problem with the objective of predicting words that surround a given word in a document. While CBOW is computationally faster, Skip-gram produces better vector quality, especially for infrequent words [31].

Several varieties of Word2Vec have been researches, such as Item2Vec [32], where instead of words, items are clustered together, and Doc2Vec [33], where the embeddings are learnt at the document level.

Chapter 3

Methodology

The purpose of this chapter is to provide an overview of the research method, paradigm, dataset as well as the evaluation methods used in this thesis. Section 3.1 describes the algorithms and methods used. Section 3.2 details the research paradigm. Section 3.3 focuses on the data collection and sampling. Section 3.4 Finally, Section 3.5 describes the framework selected to evaluate the trained embedding models.

3.1 Algorithms

3.1.1 Word2Vec: Skip-gram

As already mentioned, the choice in this project for evaluating Word2Vec in gossip learning environment is due Word2Vec's popularity and widespread use as well as the lack of prior research.

And while CBOW is slightly faster computationally than Skip-gram, predicting context words is more intuitive in the sense that it is interesting for the purpose of this project to find out the context words from a target word than otherwise; it is also easier to perform a qualitative evaluation and understand it with Skip-gram. Additionally, extensions that improve the quality of the vectors and the speed of training, such as using negative sampling [6] have been presented. Therefore, Skip-gram architecture, as shown in Figure 3.1, is the architecture of choice in this project.

In terms of architecture, Skip-gram is a shallow neural network with only one hidden layer. The input of the network is a one-hot encoder while the output is a softmax function or the probability distribution over all words in the vocabulary; in other words, the output probability distribution is the likelihood

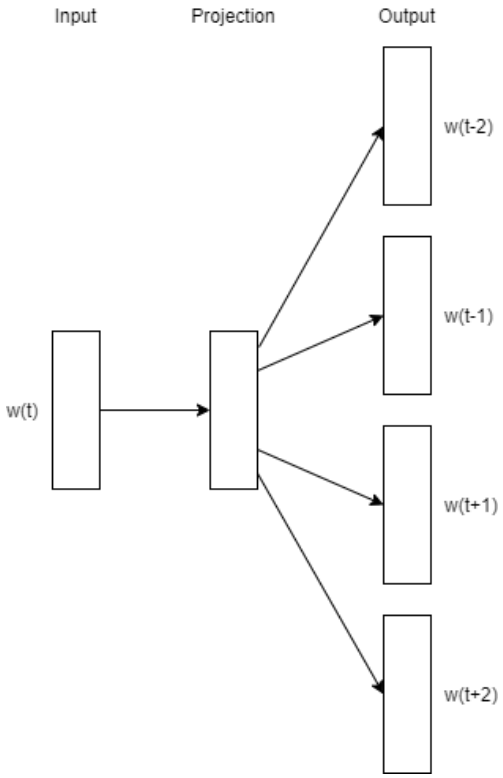

Figure 3.1: The Skip-gram architecture. Skip-gram's training objective is to learn representations that predict nearby words well.

of a word being selected as the context of the target word. The weight matrix of the hidden layer has a dimension of $W \times d$, where W is the size of the vocabulary and d is the number of neurons in the hidden layer - also known as the embedding size. These embeddings are called *continuous* d-dimensional vector representations, or embeddings of size d.

Therefore, as per this architecture, there will be two matrices of the same the $W \times d$ dimension in Word2Vec in general: the embedding weight matrix from the hidden layer, M_1 and word vector matrix that is the lookup table, M_2.

A way of representing the words as vectors is to use one-hot encoding; an integer index is assigned to $w \in \{1, ..., W\}$ which represents each word with a one-hot vector of its index - the rest of the words are represented by 0. However, the issue with one-hot encoding is that it does not preserve the semantic meaning of the word and that with more words, the size of it can be inefficient for storage.

Continuous vectors are vector representations of words using real continuous

numbers instead of the binary one-hot encoding. The length of such vectors correspond to the embedding's dimension d.

As the goal of training a Skip-gram model is to predict near context words that surround a target words, given a corpus of text containing W unique words as a sequence of $w_1, ..., w_t$ of T total words, Skip-gram aims to maximize the following average log probability

$$\frac{1}{T}\sum_{t=1}^{T}\sum_{-c\leq j\leq c, j\neq 0}\log p(w_{t+j}|w_t) \tag{3.1}$$

where c determines how many positions back and ahead to consider in order to collect the context of the target word. In other words, the objective is to maximize the probability of w_t being predicted as the context of w_{t+j} for all training pairs.

Generating target-context word training pair samples is done by having pairs of target-context word pairs as per given *window size* or *span*. Figure 3.2 shows the creation of some of these training pairs for an example text "The train goes backward through the tunnel". A span is twice the size of a *skip window* plus one, which is the target word itself.

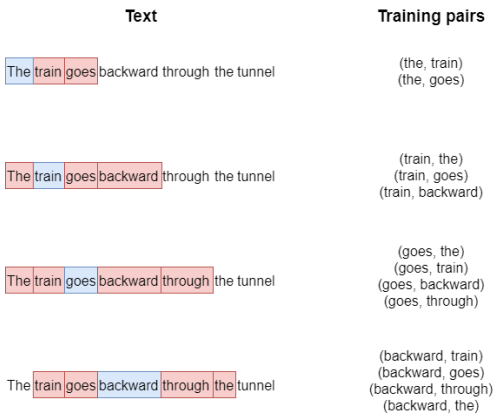

Figure 3.2: The creation of training pairs for the Skip-gram model.

3.1.2 Softmax Function

The softmax function that defines $\log p(w_{t+j}|w_t)$ is

$$p(w_O|w_I) = \frac{\exp({v'_{w_O}}^T v_{w_I})}{\sum_{w=1}^{W}\exp({v'_w}^T v_{w_I})} \tag{3.2}$$

where v_w and ${v'}_w$ are the input and output vector representations of w respectively, and W is the size of the vocabulary. Using softmax however, is computationally expensive, because the cost of computing $\nabla \log(p(w_O|w_I)$ is dependent and proportional to W as it requires scanning through the output embedding of all words in the vocabulary, and the size of such vocabularies can be huge, in the range of hundreds of thousands or even millions.

Hierarchical softmax [34] is an efficient approximation of the full softmax. Instead of evaluating the output embedding to obtain the probability distribution, hierarchical softmax uses a binary tree representation of the output layer with the W words as its leaves. As such, hierarchical softmax needs to evaluate only about $\log_2(W)$ nodes.

3.1.3 Noise Contrastive Estimation

An alternative to hierarchical softmax is using the negative sampling method: Negative Contrastive Estimation (NCE) [35]. The idea behind NCE is to differentiate data from noise with logistic regression. More specifically, the model learns to differentiate between the correct and incorrect randomly generated training pairs. A hyperparameter m, which is the number negative samples, determines the number of incorrect pairs drawn for each correct pair. All the negative samples have the same input word w_I as the original training pair , while the output word w_O is randomly drawn from a random noise distribution.

If we define D as the set of all correct pairs and D' as the set of the negatively sampled pairs, the objective of NCE is to maximize the following objective

$$\sum_{(w_I,w_O)\in D} \log \sigma({v'_{w_O}}^T v_{w_I}) + \sum_{(w_I,w_O)\in D'} \log \sigma(-{v'_{w_O}}^T v_{w_I}) \tag{3.3}$$

3.1.4 Gossip Learning

The gossip learning protocol implemented in this project is similar to the one presented in Algorithm 2.3 with a few adjustments. Algorithm 3.1 shows the flow of the gossip protocol implemented and reflects the simulation of the gossip protocol used in this project rather than the more abstract Algorithm 2.3. This is in part to facilitate the replication of the implementation. The implemented algorithm, as mentioned, assumes that all nodes have similar computation and communication speeds, so that asynchronous communications

can be accurately approximated with synchronous message passing.

Algorithm 3.1: Implemented Gossip Learning Algorithm

$currentModel \leftarrow$ INITMODEL()
$lastModel \leftarrow currentModel$
loop
  $p \leftarrow$ RANDOMPEER()
  if *# connecting peers of* $p > 2$ **then**
    Drop peers randomly until # connecting peers = 2
  SEND$(p, currentModel(s))$
end loop
procedure ONMODELRECEIVEDBYDEVICE(m_1, m_2)
  $currentModel \leftarrow$ CREATEMODEL(m_1, m_2)
end procedure
function CREATEMODEL(m_1, m_2)
  if m_1 *and* m_2 **then**
    $m_{rec1} = m_1$
    $m_{rec2} = m_2$
    $lastModel \leftarrow$ either m_1 or m_2 at random
  else if m_1 *and* m_2 *is None* **then**
    $m_{rec1} = m_1$
    $m_{rec2} = lastModel$
    $lastModel \leftarrow m1$
  else if m_1 *is None and* m_2 *is None* **then**
    $m_{rec1} = lastModel$
    $m_{rec2} = lastModel$
  $merged =$ MERGE(m_rec1, m_rec2)
  loop
    $updated \leftarrow$ UPDATE($merged$)
  end loop
  return $updated$
end function

After initialization, once the algorithm enters into the loop of the main gossip protocol, each node, which is willing and capable of contacting others at random, chooses a target peer randomly in the network. For each peer p targeted, if it is connected to by more than two nodes, then each of the connecting nodes is randomly dropped until there are only two nodes connecting. In an asynchronous communications setting, the first two connecting nodes will be the peers for any given node. The reason why

additional models received are randomly dropped is that this increases the likelihood the exchange of models is balanced in the network. Once each node has been assigned a peer or drops its communication, the exchange of information occurs. Each of the two possible m_1 and m_2 refers to the pair of matrices M_1 and M_2, and only in one of the settings of the experiments where M_2 only was exchanged.

Upon receiving (or not receiving) information from the connecting peers, as detailed by the function CreateModel, the received models m_{rec1}, m_{rec2} are merged by averaging and then updated on the local dataset. However, whether the node receives from one, two, or no peers determines the assignment of the values of m_{rec1} and m_{rec2}. If a node receives from two peers, then m_{rec1} and m_{rec2} are assigned the values of the received models. If a node receives from only one peer, then either m_{rec1} or m_{rec2} is assigned the value from that peer, while the other is taken from the last received model, and if it does not receive from any peer, then it uses the last stored model. This is done locally as defined before another round of information exchange occurs.

Furthermore, these assumptions are taken into consideration

- the data cannot be removed, therefore the models need to visit the data instead, hence the need for a data-private configuration'
- the reliability of the communication between the nodes within the network is not considered and is assumed robust;
- the nodes are assumed to have similar computation and communication speeds and capabilities as using a synchronized simulation is a good approximation when you know that the asynchronicity is minimal, which is the case if you set the computation and communication speeds to be similar;
- the contents of the nodes agree on a set of vocabulary beforehand and no new words are added to the vocabulary afterwards.

3.2 Research Paradigm

From an epistemological point of view, this research leans towards the positivist end of the spectrum, relying on the observed data from the experiments run. However, there is a room for constructivism as pertains to some of the liberties taken in terms of approach and methodological choices. In addition, the qualitative evaluation to be done breaks this research away from the rigid

positivism approach. In other words, this research will mostly follow an experimental paradigm, while the analysis will include a mix of quantitative and qualitative techniques.

This is due to the nature of the research of NLP where researches have not agreed on a universal method of evaluation and quality assessment. Further, the purpose of this project influences some of the choices made within this project approach and experimentation.

This project is also deductive in nature. That being said, another important aspect of this research is its iterative nature, whereby subsequent experiments and analysis were driven partially by the final goals and partially by the findings of the previous iterations.

3.3 Data Collection

In line of the purpose of the project and under the recommendation of the project owners, the main dataset used is the Wikipedia articles dump [36] of more than 16GB. The dump contains more than 6 million articles and it is in the form of wikitext source with embedded XML metadata.

The choice of using Wikipedia articles instead for instance Twitter dump is that Wikipedia articles are well-structured and slightly formal, as opposed to the more informal, speech-like nature of social media posts. For this reason, Wikipedia articles resemble the typical corpora of the organizations in our target scenario that was presented in the introduction. However, there are more basic reasons to consider; another important reason is that Wikipedia datasets are very often used in the NLP literature for pre-training large models and they are well known to produce high quality results. Additionally, Wikipedia dump data is relatively easy to preprocess.

3.3.1 Sampling

The experiments in this project test the effects that two orthogonal characteristics of the corpora have on the produced embeddings. The first characteristic is the size distribution, while the second is the topic distribution. To test the former, experiments are run with both almost-equally-sized corpora and with highly skewed corpora sizes. To test the latter, some experiments are run with Wikipedia contents assigned randomly to the nodes, while in other cases the assignment is done so that each node has one topic that is predominant in its corpus, while others are under-represented.

For this reason, the extraction and preprocessing of the dataset are required. A tool based on WikiExtractor [37] is used to extract and clean text from the Wikipedia data dump. Depending on the arrangement of the content of the nodes (elaborated further in Chapter 4), the tool extracts datasets of certain sizes and topicality. For instance, for datasets of a certain topic, another tool that scours the list of articles based on a given category or categories and depth of subcategories, PetScan [38] is used. This tool retrieves the list of subcategories of the given depth and then this list is fed to WikiExtractor as an input, which in turn extracts the articles belonging to those subcategories.

Further down the pipeline, the extracted corpora is preprocessed using Natural Language Toolkit (NLTK). In this process, the text is prepared for the next steps of learning. The stop words are also summarily removed from the corpora. The reason for removing stop words is twofold: first, from running experiments with and without stop words, the models converge quicker when stop words are removed. Second, for the purpose of this project, the architecture used, and the NLP task, removing stop words does not negatively affect any semantic context. The list of the English stop words are given in Appendix A.

3.4 Experimental Setup

3.4.1 Hardware and Software Used

The hardware used for the purpose of training the models in this project is of the same specifications throughout. This ensures that the results in terms of time taken, model quality are comparable. The experiments whose results are presented or alluded to in this report were run on a machine with an Intel Xeon 4214 with 12 cores (24 threads), 2.2 GHz base clock, 3.2 GHz max turbo boos, and a 16.5 MB cache. The GPU used is an NVidia Quadro RTX 5000, with 3072 CUDA cores, and around 16 GB GDDR6 usable GPU memory.

The experiments are executed in Python scripts that uses both the CPU and GPU resources for training, with emphasis on using the GPU for the learning part. The development is originally done on Jupyter notebook and then converted into script that takes in different arguments.

The choice of using Python is due to the availability of libraries and implementations for Python that are useful for carrying out this project.

3.4.2 Libraries Used

The following is a list of libraries used for training the models and generating results. Not all libraries are required to reproduce the training of the models; some are used to visualize the results that can be interpreted in a meaningful way. Some of the libraries mentioned will be accompanied by a brief explanation of how they are used.

The libraries used are as following: `argparse`, `collections`, `dill` for pickling models and data, `gensim` to download and process the pre-trained Word2Vec model, `logging`, `math`, `matplotlib`, `networkx` for plotting a graph of the information exchange between nodes, `nltk`, `numpy`, `os`, `random`, `string`, `tensorflow`, `time`, `traceback`, `urllib`, and `zipfile`.

3.5 Evaluation framework

Evaluation methods for embeddings are generally categorized into two, namely intrinsic and extrinsic evaluations [39]. Extrinsic evaluators use word embeddings as input features to a specific NLP task in order to measure the embedding quality, and are usually computationally expensive.

Intrinsic evaluators on the other hand, are independent of any NLP task. One such evaluation method is word similarity, which correlates the distance between word vectors. The goal of this evaluator is to measure how well the word vector representations capture the similarity perceived by humans. A commonly used evaluator, which is also used in this project is the cosine similarity

$$\cos(w_x, w_y) = \frac{w_x \cdot w_y}{||w_x|| \cdot ||w_y||} \tag{3.4}$$

where w_x and w_y are two word vectors and $||w_x|| \cdot ||w_y||$ is the l_2 norm. The normalization of the vector length makes it scalable and robust. This evaluator is thus used to determine the nearest context words for each word of the target words, also referred to as evaluation words.

Additionally, another metric to measure the quality of the word embeddings is generated by comparing the nearest predicted context words for each of the target words from the trained embeddings in this project and the n nearest context words for the same target words from a pre-trained model. The pre-trained model used for comparison is the Google News dataset of similar embedding size ($d = 300$) and contains a vocabulary of 3 million words

[40]. The embedding was trained with Word2Vec using negative sampling and therefore is comparable in terms of architecture and algorithm. Additionally, for the purpose of the comparison, the vocabulary set has been matched with the that used in the experiments.

If we define this metric as Word2Vec similarity $w2v_{sim}$, then it would be given as

$$w2v_{sim} = \sum_{w_e \in W_e} \sum_{w_{tw} \in W_{int}} sim(w_{tw}) \tag{3.5}$$

where w_e is an evaluation word in the evaluation words set W_e, w_{tw} is a word in the set W_{topn}, which is a set of the top n-most similar words to each $w_e \in W_e$ from the pre-trained Word2Vec model, and W_{int} is at set which contains the intersection of the nearest neighbors of $w_e \in W_e$ generated by the model, and the n-most similar words from the pre-trained Word2Vec model. $w2v_{sim}$ also helps to show how fast each model converges before saturation of the nearest context words generated.

Moreover, the *loss* which is the average NCE loss for each batch, is used for evaluation.

Furthermore, in order to evaluate the embedding models, because of the topicality of some of the nodes, it would be interesting to qualitatively assess the context words with respect to the given target words.

Comparative results using these evaluation methods are shown in Chapter 5.

Chapter 4

Experiments

This chapter describes the experiments and evaluation methods run as outlined in Chapter 3. This chapter further elaborates the different configurations tested in pursuit of the purpose and goals of this project. Section 4.1 details the general parameters used for running the different configurations, while Section 4.2 describes the different configurations under which the experiments ran. Chapter 5 shows the results corresponding to each of these configurations, as well as the analysis and discussion based on them.

4.1 Simulation Model Parameters

With the gossip learning protocol implemented in this project, there are two major iterations that are performed, one inside the other. The outside loop, as shown in Algorithm 3.1, is completed when the peers are selected, the information exchanged, and local merging and update complete. Let the number of iterations for this loop be $GLsteps$. The inner loop in the CreateModel function, refers to how many times after the merging the update or training happens locally. Let this number of training locally before the next information exchange round be called $localsteps$. Each step of the local training uses one $batch$. Therefore, using the gossip learning protocol, the total number of batches processed is $GLsteps \times localsteps \times$ the number of nodes.

As already mentioned, finding the best hyperparameters or training the best possible models were not the goal of this thesis. Therefore, although the choice hyperparameters used in the experiments have been mildly tuned, they are not by any means the best possible parameters.

To maintain comparability, in both the traditional centralized case and the

gossip learning case, the total number of batches is equalized, or $totalbatches$ is set to $5,000,000$. The batch size itself is set to 2048 The number of nodes in distributed cases is set to 10. The embedding size or d is 200. The maximum vocabulary size for each local node is set to $200,000$ and only words that appear more than 10 times are included in the vocabulary. The *window size* or *span* is 25, while the number of negative samples used for the NCE is 64.

Finally, the learning rate used in the training is a decaying learning rate with the initial value of $\alpha = 0.7$ and decays according to the following

$$learningrate = \begin{cases} min(\alpha, \log_{10} \frac{counter}{0.998 \times totalbatches}), & learningrate \geq 0 \\ 0, & \text{otherwise} \end{cases} \tag{4.1}$$

where $counter$ refers to the times the local node has run one training of iteration since the start. Figure 5.1 shows the learning rate decay.

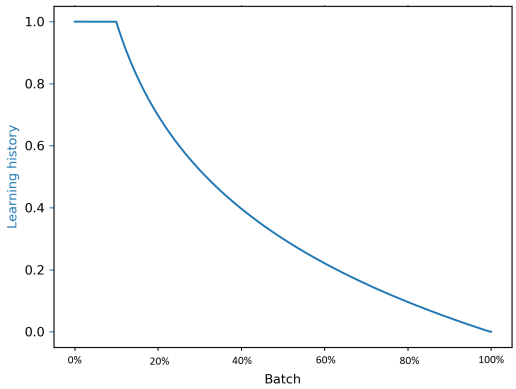

Figure 4.1: Learning rate history.

While the *loss*, that is the average NCE loss for the $batch$ is calculated and logged at every iteration of $localsteps$, $w2v_{sim}$ is measured and logged every $1,000$ $localsteps$ because it is more computationally expensive, and tends not to change significantly.

The evaluation words used for all cases are ["five", "war", "work", "year", "people", "location", "october", "state", "science", "rights", "history", "money", "bank", "man", "woman", "growth", "spring", "life", "hard", "culture", "medium", "family", "alien"]. These are chosen so based on the topicality of the nodes in the case where the nodes are heterogeneous. For each of the evaluation words,

eight of the nearest context words as per the cosine similarity evaluator are shown in the results.

When testing the effects of having a different, domain specific corpus at each node, the Wikipedia dataset was split according to the following categories: *science*, *politics*, *business*, *humanities*, and *history*.

Beyond these, the parameters and settings are explained in each configuration in the next section.

4.2 Setup and Implementation

4.2.1 Traditional Centralized Training

To establish the baseline to compare to, the first experiment configuration is on the traditional non-distributed, centralized configuration of Word2Vec. In this setup, the datasets used are also drawn by topic of almost equal sizes. Each of the topics *science*, *politics*, *business*, *humanities*, and *history* constitute a relatively equal part of the dataset.

4.2.2 Gossip Learning with Frequent Exchange

In this configuration, Word2Vec is run on gossip learning with frequent exchange of models. To be more specific, $GLsteps = 50,000$ and $localsteps = 10$. In other words, The nodes exchange their models every 10 batches, thus performing a total 500,000 exchanges during the whole 5,000,000 training batches.

This configuration means that the models in each node are merged very frequently and thus are likely to be closely similar to each other. However, the downside of the frequent communication is that it is communication-intensive, thus requiring a large bandwidth between the nodes and potentially driving up cost. And in real-world scenarios, reducing communication is of importance, which would be addressed in the next subsection.

Four different sub-configurations of experiments in this configuration are run:

- *topic-wise*: each node's corpus is of similar size and the content is divided by the five topics *science*, *politics*, *business*, *humanities*, and *history* (in that order; so nodes 1 and 2 are nodes with *science* data), thus, each topic's datasets are assigned to two nodes;

- *random balanced*: each node's corpus is of similar size but the content is drawn randomly from the whole Wikipedia articles dump;
- *random imbalanced*: each node's corpus is randomly drawn from the Wikipedia articles dump and the content sizes for the nodes are lopsided, and in some cases the ratio of content size between some two nodes is up to 1:4. However, the total size of the contents of all the nodes are similar to the total size of the contents size in the other sub-configurations;
- *half*: each node's corpus is of similar size and the content is divided by the five topics, but only only the matrix M_2 as detailed in Algorithm 2.4 is exchanged: this is an attempt to reduce bandwidth by exchanging part of the model.

The intuition behind dividing the nodes contents by topic is that often times the corpora of organizations tend to be within a specific domain, which is why it is the case in most configurations. Investigating how the models evaluate if the contents of each node are heterogeneous in terms of topic is however, also of interest, as it can be the case that some node owners own a corpus of different topics.

Setting imbalanced content sizes in one of the sub-configurations can provide insight into how the learning learning is affected when some nodes have significantly bigger corpora than others. This research direction is extremely relevant to the practical applicability of this project, as in real-world scenarios it is likely that some organizations are in possession of larger amounts of text than others.

Sending the M_2 matrix exclusively is also an interesting direction to investigate, as it has already been mentioned that limiting the overall bandwidth needed is important to make the training fast and potentially cheap. Although it is not the only optimization technique that can be employed, it is one which is related to the architecture of decentralized machine learning. Therefore, this optimization attempt is not mutually exclusive with other techniques as will be explained in Chapter 6.

4.2.3 Gossip Learning with Infrequent Exchange

This configuration mirrors the previous one, having the same 4 sub-configurations based on different data distribution. However, the model exchange frequency is reduced by a factor of 50. This is driven by the need to reduce communication frequency as it costs bandwidth and in real-world scenarios, the more frequent

the more likely things could go wrong. The parameters used in this case are $GLsteps = 1,000$ and $localsteps = 500$ instead, i.e. after each round of $1,000$ total iterations of information exchange, the local model is merged with the received and updated for 500 iterations.

The communications are reduced 50 times in this configuration comparatively, from 50,000 to 1,000. This means that each node will perform local training for 500 iterations (from previously 10) before entering another round of information exchange.

4.2.4 Local Nodes Training with Common Vocabulary

Similar to the traditional centralized approach, in this configuration, each node trains its model locally using its own datasets only, but still using a common vocabulary list of all the nodes. The common vocabulary is also drawn from the categories of *science*, *politics*, *business*, *humanities*, and *history*. The common vocabulary is required here in order to be able to perform meaningful comparisons with the previous configurations.

This is to see how the models will learn if there is no information exchange. Also, intuitively, since the contents of the nodes are divided by topic, then one hypothesis is that different models trained using different datasets may associate different context words for some ambiguous target words.

Chapter 5

Results and Analysis

In this chapter, the results of the experiments of different configurations delineated in Chapter 4 are presented and then analyzed. Furthermore, proposed solutions based on the results are given when appropriate. Then, Section 5.5 discusses these findings with respect to each other.

5.1 Traditional Centralized Training

The first experiment consists of training a simple Word2Vec model in a traditional, single-machine setting. This will serve as a baseline to understand the behaviour of the gossip-based approach. Figure 5.1 shows the $loss$ and $w2v_{sim}$ (as defined in Equation 3.5) in shows the evolution of the loss and $w2v_{sim}$ throughout the training. The figure shows a rapid decline in the $loss$ value and virtually flattens to almost zero. The $w2v_{sim}$ value can be seen to fluctuate between 55 and 70 at 1,000,000 batches and converges towards the range of between 60 and 70 towards 5,000,000 - with the baseline $w2v_{sim}$ value resting at 65. As the learning rate gets lower, the nodes apply less and less modifications to the models they receive. Thus, the models tend to become more and more similar and change less and less. So, there is less variability and less fluctuations. This might be the explanation of why the fluctuations become lower in magnitude but seem to be centered around the same value. It may not the model getting better, but rather the fluctuations are artificially reduced by reducing the learning rate. However, this is desired, i.e. a stable model. However, by making it stable some quality is lost compared to the best peak. This is likely the reason why in the original gossip learning formulation by Ormándi et al. [4], each node keeps a buffer of the last k models it received, and uses them as an ensemble to perform prediction. This window of k models

helps smoothing any fluctuations in the first part of training, when the learning rate is still high.

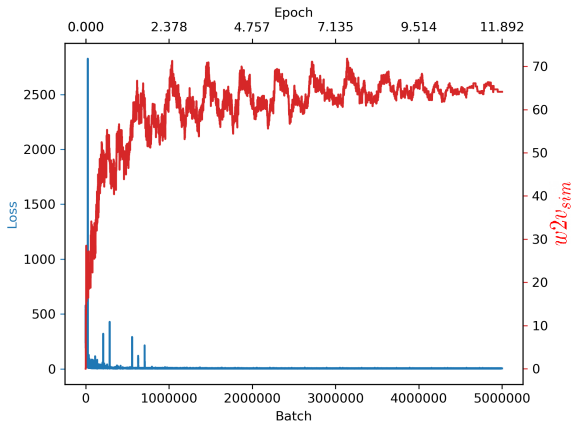

Figure 5.1: Traditional centralized learning results: $loss$ and $w2v_{sim}$ values over batches. The $w2v_{sim}$ value converges to 65.

Table 5.1 shows the target words and their corresponding eight nearest predicted context words for traditional centralized training. The qualitative judgement of these predicted context words would be interesting to juxtapose against the predicted context words in other configurations. Words designated as "UNK" is just the token used for out-of-vocabulary words.

5.2 Gossip Learning with Frequent Exchange

This section presents the findings of the gossip learning with frequent exchange configuration. Figure 5.2 shows the $loss$ values over the batches for all ten nodes for the first sub-configuration *topicwise*. The figure shows that for each of the nodes, the $loss$ values begin to flatten around 200,000 local batches.

Table 5.1: Target words and predicted context words for traditional centralized training at batch 5,000,000.

Target word	**Nearest predicted context words**
five	six, eight, four, three, nine, ten, twenty, twice
war	allied, invasion, occupation, aftermath, wars, military, fought, ally
work	interested, devoted, working, presented, conceived, noted, contributions, setting
year	years, ten, month, reached, spent, twenty, 2000, 14
people	reasons, among, despite, others, seeking, amongst, encouraged, alone
location	locations, connected, main, travel, extended, entire, open, proximity
october	september, november, april, december, february, june, march, july
state	establishing, governed, establish, representative, independent, constituted, establishment, authority
science	scientific, sciences, academic, contributions, research, studying, disciplines, thesis
rights	legal, freedom, justice, citizen, laws, legislation, act, enforced
history	historical, origins, devoted, contemporary, traced, influential, modern, influenced
money	paying, pay, paid, buy, credit, payment, buying, cash
bank	banks, capital, trading, funds, banking, shares, invested, exchange
man	says, mans, tells, knew, love, tell, strange, fate
woman	mother, man, husband, couple, girl, sisters, daughters, fathers
growth	increasing, increase, decreasing, rapidly, significantly, dramatically, rapid, increased
spring	autumn, summer, winter, fall, reaching, around, occasional, laid
life	past, yet, others, brings, believed, thought, concludes, seems
hard	easily, easy, like, make, makes, unlike, placing, making
culture	cultural, influences, influenced, cultures, contemporary, origins, historical, history
medium	typical, component, easily, consist, continuous, variations, combination, used
family	belonging, described, found, genus, UNK, belongs, belong, species
alien	encounter, encounters, hope, reveals, fate, seeing, never, strange

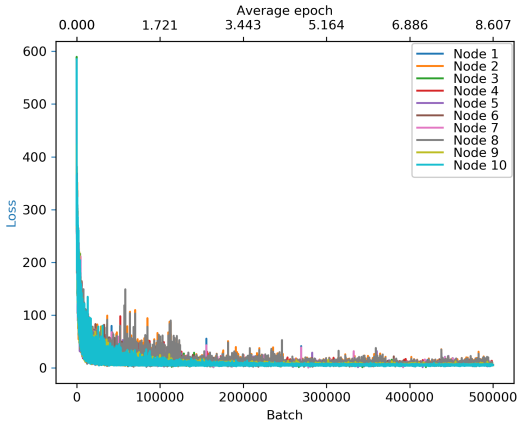

Figure 5.2: Gossip learning with frequent exchange (*topicwise*) results: $loss$ values over local batches.

Figure 5.3 on the other hand shows the corresponding $w2v_{sim}$ values over the local batches of 500,000. As the figure shows, the value of $w2v_{sim}$ starts to converge to 60 after 400,000 local batches trained. This suggests that further training will not likely yield as much improvement. Compared to the traditional centralized training configuration where the value converges to 65, in this case the value converges to 60 and this is an indication of decrease of model quality as per $w2v_{sim}$. $w2v_{sim}$ in this case does not fluctuate as it did in the traditional centralized configuration. This means that the models stabilized sooner and earlier than is the case with traditional centralized learning.

Figure 5.4 shows a graph representing the number of directed sending of models across the network. 419,487 instances of sharing occurred. As the instances of models sharing in this configuration remain similar throughout the sub-configurations, Figure 5.4 represents the other sub-configurations as well.

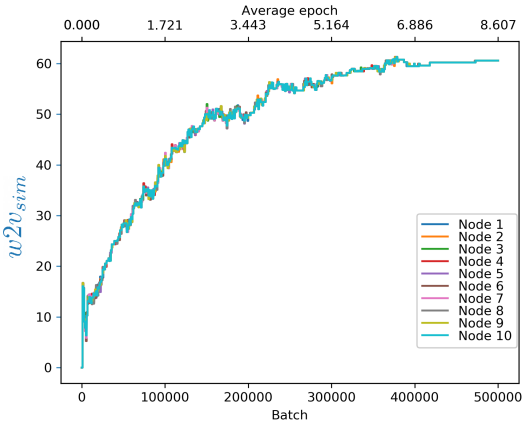

Figure 5.3: Gossip learning with frequent exchange (*topicwise*) results: $w2v_{sim}$ values over local batches. The $w2v_{sim}$ value converges to 60. In comparison, the baseline is 65.

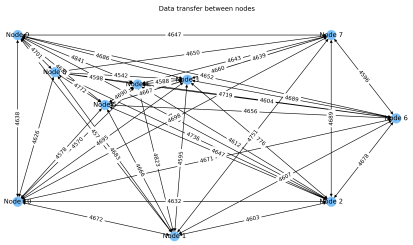

Figure 5.4: Gossip learning with frequent exchange (*topicwise*) model sharing.

Table 5.2 shows the target words and their corresponding eight nearest predicted context words for the gossip learning with frequent exchange configuration, with the *topicwise* sub-configuration. As the frequency of model sharing is high, the models for all 10 nodes practically predict the same context words. Therefore, for tables detailing the predicted and context words in this section, one table represents all 10 nodes.

As for qualitative judgement of the context words shown in Table 5.1 and Table 5.2, in the latter, there is a lack of capturing of semantic context in some target words such as *spring* and *money* where most of the context words predicted for these words are not semantically related.

Table 5.2: Target words and predicted context words for gossip learning with frequent exchange *topicwise* at each local batch 500,000.

Target word	**Nearest predicted context words**
five	four, three, two, six, last, one, times, first
war	ii, military, british, led, german, took, world, army
work	created, others, based, way, works, become, see, well
year	years, five, last, day, four, second, first, times
people	among, according, others, become, even, many, however, though
location	main, still, well, also, several, around, along, part
october	september, april, march, july, june, december, january, november
state	government, general, local, country, established, led, according, national
science	research, work, scientific, society, history, studies, based, development
rights	act, law, legal, stated, public, state, government, right
history	modern, among, early, work, works, culture, according, society
money	would, could, time, take, make, without, made, order
bank	capital, local, country, major, largest, established, part, business
man	wrote, said, death, life, never, upon, another, others
woman	man, men, life, death, never, others, young, another
growth	low, increased, due, increase, less, significant, high, higher
spring	around, summer, along, three, near, four, times, two
life	others, become, even, upon, thought, way, see, though
hard	either, similar, like, much, make, possible, example, different
culture	cultural, history, modern, traditional, among, society, important, many
medium	material, form, much, similar, generally, either, large, usually
family	found, described, species, genus, known, native, UNK, southern
alien	time, making, times, set, could, since, interest, another

Figure 5.5 shows the $loss$ values over the batches for all nodes for the second sub-configuration *randombalanced*, where data is distributed randomly among the nodes, with evenly-sized corpora. For each of the nodes, the $loss$ values begin to flatten around 300,000 local batches, compared to 200,000 in the previous case.

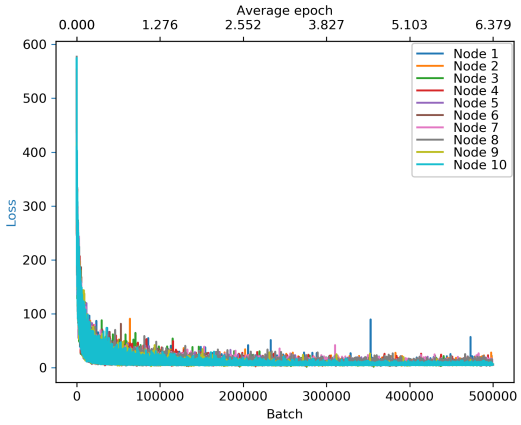

Figure 5.5: Gossip learning with frequent exchange (*randombalanced*) results: $loss$ values over local batches.

Figure 5.6 shows the corresponding $w2v_{sim}$ values over the local batches trained. The value of $w2v_{sim}$ converges after 400,000 batches of training as well at the value of 60. This seems to indicate that the topic-distribution of the dataset does not affect the quality of the word embedding overall. It is interesting to see whether this phenomenon also holds true when more infrequent exchanges are involved.

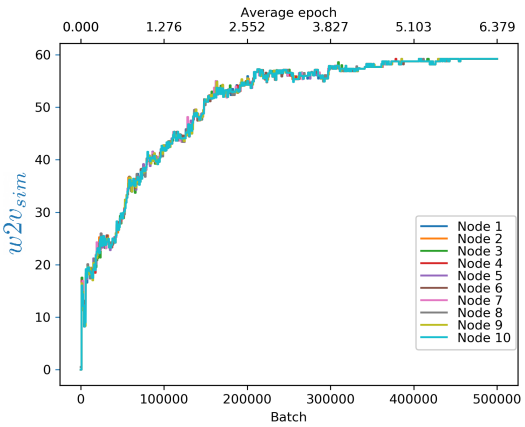

Figure 5.6: Gossip learning with frequent exchange (*randombalanced*) results: $w2v_{sim}$ values over local batches. The $w2v_{sim}$ value converges to 60. In comparison, the baseline is 65.

Table 5.3 shows the evaluation words and their corresponding eight nearest predicted context words for the second sub-configuration. The qualitative observation made with regards to Table 5.2 also applies for this particular experiment. This suggest that gossip learning finds it difficult to capture certain patterns compared to tradition centrealized learning. Therefore, it is interesting for future research to investigate how each node can learn from the corpora of others in the network while still retaining the semantic nuances based on its local contents.

Figure 5.7 shows the $loss$ values over the batches for all nodes for the third sub-configuration *randomimbalanced*, where data is distributed randomly among the nodes, with skewed-sized corpora. As the figure shows, for each of the $loss$ values, the $loss$ values begin to flatten around 200,000 local batches.

Table 5.3: Target words and predicted context words for gossip learning with frequent exchange *randombalanced* at each local batch 500,000.

Target word	**Nearest predicted context words**
five	three, four, six, two, seven, last, first, one
war	ii, military, army, british, german, forces, battle, french
work	working, works, life, various, well, others, several, among
year	first, started, five, years, six, went, three, following
people	according, considered, many, among, others, important, although, within
location	main, site, small, around, large, along, across, area
october	september, april, november, june, december, february, july, august
state	established, public, community, city, office, current, local, national
science	institute, research, studies, study, university, society, program, technology
rights	human, act, political, social, government, case, support, women
history	among, including, since, work, several, also, times, well
money	would, make, take, help, without, said, never, could
bank	largest, established, local, business, central, capital, part, company
man	never, life, young, said, good, story, together, another
woman	man, young, life, said, never, death, see, together
growth	development, important, developed, within, use, form, low, large
spring	one, two, three, since, following, part, five, also
life	young, others, together, work, man, become, wrote, said
hard	another, set, one, like, right, way, little, see
culture	among, many, important, work, works, various, modern, others
medium	many, important, work, several, various, modern, works, others
family	known, father, name, death, UNK, children, great, found
alien	present, called, way, little, much, around, still, instead

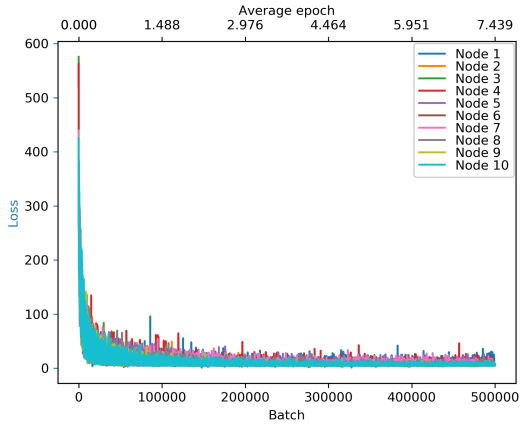

Figure 5.7: Gossip learning with frequent exchange (*randomimbalanced*) results: $loss$ values over local batches.

The corresponding $w2v_{sim}$ values over the trained local batches are shown in Figure 5.8. $w2v_{sim}$ begins to converge and plateau at 60 after 400,000 local batches a well. This seems to suggest that, like topic-distribution, size-distribution of the dataset in each node is not a factor that determines the final quality of the embeddings. These results are extremely similar compared to the findings in the last sub-configuration. It can be inferred therefore, that size-distribution of node datasets also does not determine the embedding quality.

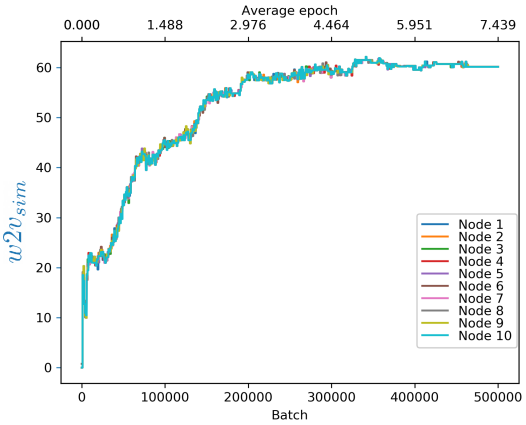

Figure 5.8: Gossip learning with frequent exchange (*randomimbalanced*) results: $w2v_{sim}$ values over local batches. The $w2v_{sim}$ value converges to 60. In comparison, the baseline is 65.

The target words and their corresponding predicted target words for the third sub-configuration are shown in Table 5.4. As the predicted context words are relatively similar in this sub-configuration and the last two, it further consolidates the suggestion that neither topic-distribution nor size-distribution of the datasets in the nodes plays a role in determining the embedding quality.

Lastly for this configuration, Figure 5.9 shows the $loss$ values over the local batches for the last sub-configuration where the datasets of the nodes are topic-divided, but only one weight matrix transmitted. The $loss$ values for some of the nodes flatten after 200,000 local batches. However, for other nodes, the $loss$ values fluctuate all the way to 500,000 batches.

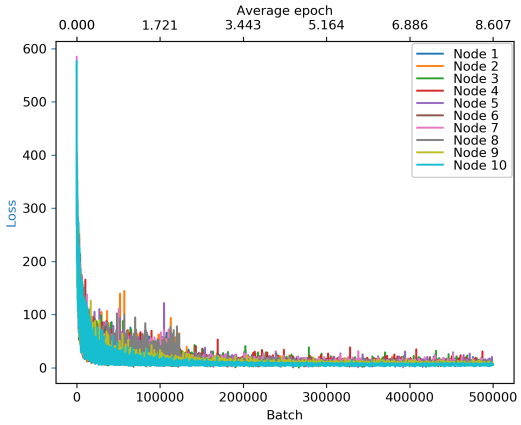

Figure 5.9: Gossip learning with frequent exchange (*half*) results: $loss$ values over local batches.

It is interesting to notice that the $w2v_{sim}$ values do not converge in this sub-configuration, as shown in Figure 5.10. The $w2v_{sim}$ values for the nodes converge to their final values between 400,000 and 500,000 local batches, with the local values ranging between 22 and 31. This is significantly lower than the $w2v_{sim}$ values seen for the other sub-configurations.

There does not seem to be any correlations between the topic of node content and its final $w2v_{sim}$ value either; case in point, nodes 9 and 10, both of which are history nodes topic-wise.

Table 5.4: Target words and predicted context words for gossip learning with frequent exchange *randomimbalanced* at each local batch 500,000.

Target word	**Nearest predicted context words**
five	four, three, six, two, seven, last, first, one
war	ii, army, military, german, forces, british, battle, sent
work	works, working, life, others, well, various, several, among
year	first, years, five, started, six, went, three, following
people	according, considered, among, many, others, given, important, within
location	main, around, along, small, large, site, outside, within
october	september, november, february, april, december, june, march, july
state	public, established, city, community, local, department, central, national
science	studies, research, institute, study, university, program, society, education
rights	act, political, human, government, social, women, members, organization
history	among, including, work, since, also, several, well, leading
money	would, without, take, make, however, become, could, said
bank	largest, established, capital, part, central, city, local, center
man	never, life, said, young, gave, story, together, good
woman	man, life, young, never, said, together, death, wrote
growth	within, important, large, common, areas, use, food, system
spring	two, three, major, end, one, time, years, next
life	others, young, together, work, man, wrote, said, see
hard	another, together, like, set, make, way, much, though
culture	among, important, various, many, work, works, modern, society
medium	developed, various, use, modern, particularly, many, work, used
family	known, father, name, children, death, UNK, together, great
alien	thus, fire, continued, period, remained, great, although, still

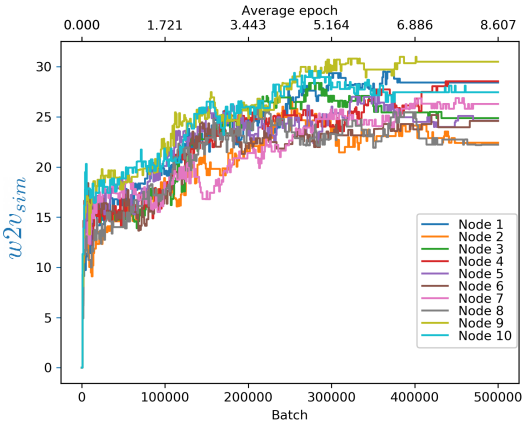

Figure 5.10: Gossip learning with frequent exchange (*half*) results: $w2v_{sim}$ values over local batches. The final $w2v_{sim}$ values range between 22 and 31. In comparison, the baseline is 65.

The non-convergence of the $w2v_{sim}$ could have been attributed to imbalanced sharing of models among the peers in the network. However, as shown Figure 5.11, this is not the case as model sharing in this sub-configuration is also balanced. 418,537 instances of partial model sharing occurred for this sub-configuration.

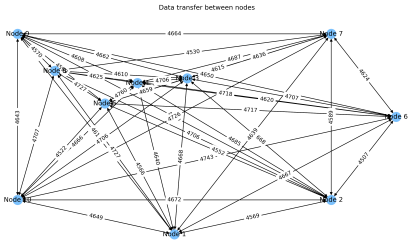

Figure 5.11: Gossip learning with frequent exchange (*half*) model sharing.

Table 5.5 shows the target and corresponding predicted context words for the *half* sub-configuration for node 9, the node with the highest final $w2v_{sim}$ value. The context words shown in the table are of lower quality semantically compared to the other sub-configurations. More training iterations are not likely to improve this result as well as Figure 5.10 suggests. Based on the

results obtained from this sub-configuration, as it is, this sub-configuration does not yield favorable results for the purpose of this project as described in Section 1.3.

Table 5.6 summarizes the results of this configuration. The training time of all the sub-configurations are comparable to each other, as well as to the traditional centralized training configuration. It must be understood however, that the training time does not take into account the time and cost it takes to share the models between the peers in the network, which depends on the size of the models and costs bandwidth. In terms of Word2Vec similarity, the first three subcategories are comparable in their results, which seems to indicate (at least within the context of the configuration) that the heterogeneity and equality of sizes of the datasets are not a determining factor. The batch around which $w2v_{sim}$ values converge are also comparable.

Table 5.6: Summary of sub-configuration results of gossip learning with frequent exchange.

Sub-configuration	**Training time (h)**	$w2v_{sim}$	**Saturation batch**
topicwise	21.625	60	400,000
randombalanced	20.074	60	400,000
randomimbalanced	22.875	60	400,000
half	20.409	22 - 31	400,000 - 500,000

5.3 Gossip Learning with Infrequent Exchange

In this configuration, the communication between the nodes in the network has been reduced by a factor of 50. This is an attempt to reduce the cost of the bandwidth used to transmit the models. Figure 5.12 shows the $loss$ values over the batches for all ten nodes for the first sub-configuration *topicwise*. The figure shows that for some of the nodes, the $loss$ values begin to flatten around 150,000 local batches, while for others, fluctuations in the value occur all the way to the end.

Table 5.5: Target words and predicted context words for gossip learning with frequent exchange *half* at each local batch 500,000.

Target word	**Nearest predicted context words**
five	three, UNK, two, called, later, new, part, known
war	became, state, world, part, government, later, states, french
work	one, time, also, however, would, well, many, made
year	first, years, new, three, two, later, since, made
people	many, however, according, one, among, although, also, well
location	north, around, area, end, although, known, part, based
october	first, new, year, years, later, since, second, two
state	states, government, world, people, part, war, became, according
science	form, often, based, may, many, one, well, another
rights	would, us, however, later, another, according, time, group
history	early, one, time, many, also, including, known, made
money	control, number, process, us, new, would, made, since
bank	area, large, part, three, known, south, UNK, two
man	time, one, later, made, called, two, well, also
woman	four, time, power, main, de, early, one, years
growth	around, may, large, small, found, since, years, several
spring	end, called, based, well, first, UNK, time, others
life	however, many, one, people, often, time, may, work
hard	even, group, could, system, found, make, used, include
culture	according, people, many, however, known, among, although, state
medium	rapid, small, came, area, important, art, music, like
family	well, known, one, according, called, years, however, time
alien	family, various, another, name, well, west, made, called

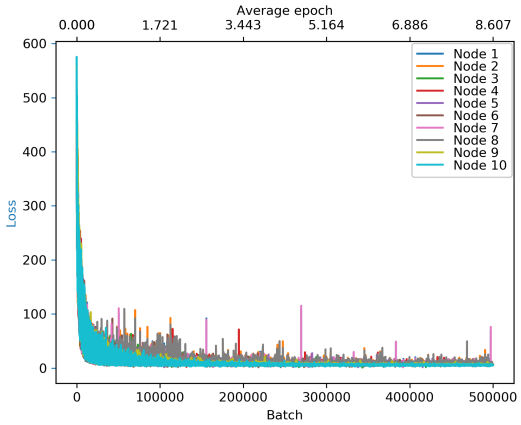

Figure 5.12: Gossip learning with infrequent exchange (*topicwise*) results: $loss$ values over local batches.

Figure 5.13 on the other hand shows the corresponding $w2v_{sim}$ values over the local batches of 500,000. As the figure shows, the value of $w2v_{sim}$ starts to converge to 62 but only after 320,000 local batches trained int this case. Further training will not likely yield as much improvement. It is interesting here to notice that $w2v_{sim}$ converges quicker (with fewer iterations) compared to the gossip learning with frequent exchange counterpart. However, this phenomenon with the infrequent exchange seems to be countered by the results of the next two sub-configurations. Therefore, it cannot be conclusively said that with more infrequent exchange, the $w2v_{sim}$ value converges sooner.

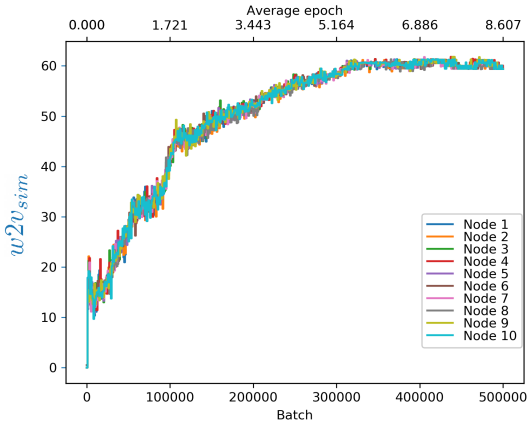

Figure 5.13: Gossip learning with infrequent exchange (*topicwise*) results: $w2v_{sim}$ values over local batches. The $w2v_{sim}$ value converges to 62. In comparison, the baseline is 65.

Figure 5.14 shows a graph representing the number of directed sending of models across the network. 8,355 instances of sharing occurred. As to be expected, the sharing of model is not as frequent as it is in the previous configuration. Likewise, as the number of instances of model sharing throughout this configuration remains relatively similar, Figure 5.14 represents the transmission of models for all the other sub-configurations.

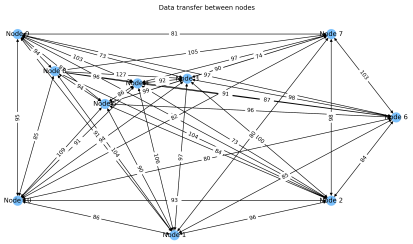

Figure 5.14: Gossip learning with infrequent exchange (*topicwise*) model sharing.

Table 5.7 shows the target words and their corresponding eight nearest predicted context words for the gossip learning with frequent exchange configuration, with the *topicwise* sub-configuration. Although the frequency of model

exchange is not as high as it is in the previous configuration, the models closely converge. Thus, the models for all 10 nodes practically predict almost identical context words. Therefore, for tables detailing the predicted and context words in this section as well, one table represents all 10 nodes. This means that a reduction in communication by a factor of 50 does not reduce the final embedding quality. It is therefore interesting for future studies to investigate how much reduction in communication affects the quality of the embeddings, and how to fine-tune this number to give the highest quality-to-communication ratio.

Figure 5.15 shows the $loss$ values over the batches for all nodes for the second sub-configuration *randombalanced*. For each of the nodes, the $loss$ values begin to flatten around 200,000 local batches.

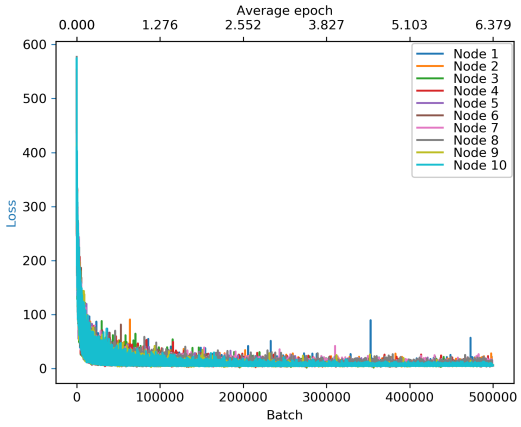

Figure 5.15: Gossip learning with infrequent exchange (*randombalanced*) results: $loss$ values over local batches.

Figure 5.16 shows the corresponding $w2v_{sim}$ values over the local batches trained. The value of $w2v_{sim}$ converges to 61 after 430,000 batches of training. Again, either with frequent or infrequent exchange, topic-distribution of node datasets does not seem to be factor determining embedding quality. In this sub-configuration however, the value of $w2v_{sim}$ converges at a later batch than in the previous sub-configuration.

Table 5.7: Target words and predicted context words for gossip learning with infrequent exchange *topicwise* at each local batch 500,000.

Target word	**Nearest predicted context words**
five	four, three, six, two, last, times, years, followed
war	ii, military, british, german, led, continued, army, took
work	created, others, based, works, various, working, become, well
year	years, five, 20, day, 15, six, last, four
people	among, according, others, become, even, many, however, despite
location	main, along, several, around, across, part, well, open
october	april, march, december, september, november, june, july, january
state	government, general, country, established, led, local, independent, national
science	research, scientific, work, society, history, works, based, developed
rights	legal, law, act, stated, organization, society, policy, us
history	modern, works, historical, early, culture, among, work, traditional
money	would, free, take, interest, could, making, working, allowed
bank	capital, country, local, largest, trade, established, major, part
man	never, wrote, said, upon, death, life, whose, john
woman	man, death, never, men, said, life, another, though
growth	increase, increased, higher, low, due, conditions, significant, levels
spring	along, around, almost, three, near, placed, two, four
life	others, thought, good, even, seen, though, see, considered
hard	either, like, similar, much, instead, generally, make, used
culture	cultural, history, traditional, modern, society, influence, among, historical
medium	generally, similar, usually, type, types, single, different, produce
family	found, species, described, genus, native, known, UNK, southern
alien	major, several, including, quickly, since, close, resulting, number

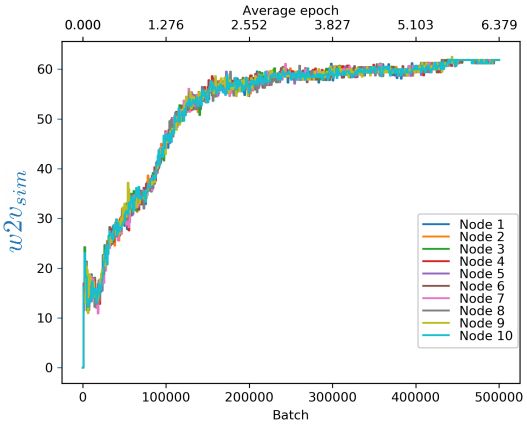

Figure 5.16: Gossip learning with infrequent exchange (*randombalanced*) results: $w2v_{sim}$ values over local batches. The $w2v_{sim}$ value converges to 61. In comparison, the baseline is 65.

Table 5.8 shows the evaluation words and their corresponding eight nearest predicted context words for the *randombalanced* sub-configuration.

Figure 5.17 shows the $loss$ values over the batches for all nodes for the third sub-configuration *randomimbalanced*. As the figure shows, for each of the $loss$ values, the $loss$ values begin to flatten around 200,000 local batches.

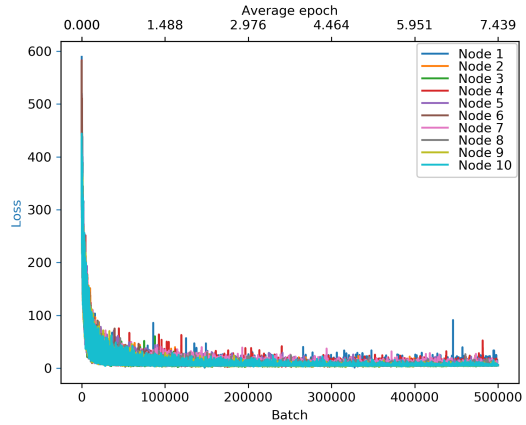

Figure 5.17: Gossip learning with infrequent exchange (*randomimbalanced*) results: $loss$ values over local batches.

The corresponding $w2v_{sim}$ values over the trained local batches are shown in Figure 5.18. $w2v_{sim}$ begins to converge and plateau at 61 after 430,000

Table 5.8: Target words and predicted context words for gossip learning with infrequent exchange *randombalanced* at each local batch 500,000.

Target word	**Nearest predicted context words**
five	three, four, six, two, one, seven, last, first
war	ii, military, army, british, german, forces, battle, french
work	works, working, well, life, several, various, many, others
year	first, five, years, following, started, three, went, four
people	according, among, many, others, considered, within, although, given
location	main, around, small, large, site, along, within, part
october	september, april, december, november, june, february, july, august
state	public, central, established, city, current, community, within, department
science	research, studies, institute, study, university, technology, society, program
rights	human, act, political, government, social, support, case, members
history	among, work, including, also, since, several, well, works
money	make, help, take, would, without, could, however, become
bank	capital, central, established, part, largest, local, city, business
man	never, life, young, said, together, good, another, little
woman	man, young, life, death, said, never, story, together
growth	within, development, developed, use, provide, large, common, result
spring	two, several, three, one, also, well, since, including
life	young, others, together, work, wrote, become, man, though
hard	like, much, make, together, never, another, though, even
culture	many, among, modern, important, various, work, works, traditional
medium	given, found, much, upon, another, called, according, see
family	father, known, whose, young, together, death, mother, children
alien	though, although, another, together, still, others, one, even

local batches a well. The results of the first three sub-configurations in this configuration indicate that the $w2v_{sim}$ value is more likely to converge later than it would in the gossip learning with frequent exchange configuration. This poses the question to the organizations wishing to train their models using gossip learning whether they prefer to achieve good embedding quality sooner but at the cost of using more bandwidth or rather the opposite, i.e. achieving comparable embedding quality later but with less communication required.

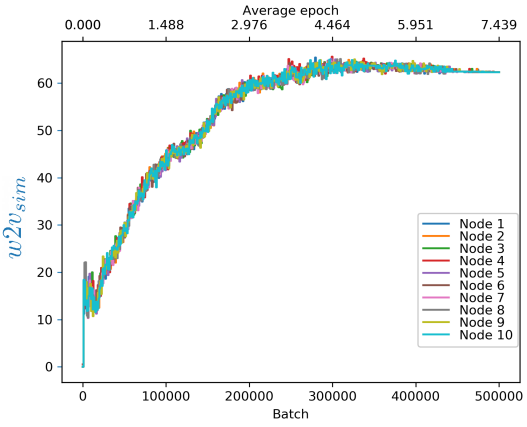

Figure 5.18: Gossip learning with infrequent exchange (*randomimbalanced*) results: $w2v_{sim}$ values over local batches. The $w2v_{sim}$ value converges to 61. In comparison, the baseline is 65.

The target words and their corresponding predicted target words for the third sub-configuration are shown in Table 5.9.

Finally, for this configuration, Figure 5.19 shows the $loss$ values over the local batches for the *half* sub-configuration. The $loss$ values for some of the nodes flatten after 200,000 local batches, while for other nodes, the $loss$ values fluctuate on the low end all the way to 500,000 batches.

Table 5.9: Target words and predicted context words for gossip learning with infrequent exchange *randomimbalanced* at each local batch 500,000.

Target word	**Nearest predicted context words**
five	four, three, six, two, seven, last, first, one
war	ii, army, military, german, forces, british, battle, sent
work	works, working, life, others, well, various, several, among
year	first, years, five, started, six, went, three, following
people	according, considered, among, many, others, given, important, within
location	main, around, along, small, large, site, outside, within
october	september, november, february, april, december, june, march, july
state	public, established, city, community, local, department, central, national
science	studies, research, institute, study, university, program, society, education
rights	act, political, human, government, social, women, members, organization
history	among, including, work, since, also, several, well, leading
money	would, without, take, make, however, become, could, said
bank	largest, established, capital, part, central, city, local, center
man	never, life, said, young, gave, story, together, good
woman	man, life, young, never, said, together, death, wrote
growth	within, important, large, common, areas, use, food, system
spring	two, three, major, end, one, time, years, next
life	others, young, together, work, man, wrote, said, see
hard	another, together, like, set, make, way, much, though
culture	among, important, various, many, work, works, modern, society
medium	developed, various, use, modern, particularly, many, work, used
family	known, father, name, children, death, UNK, together, great
alien	thus, fire, continued, period, remained, great, although, still

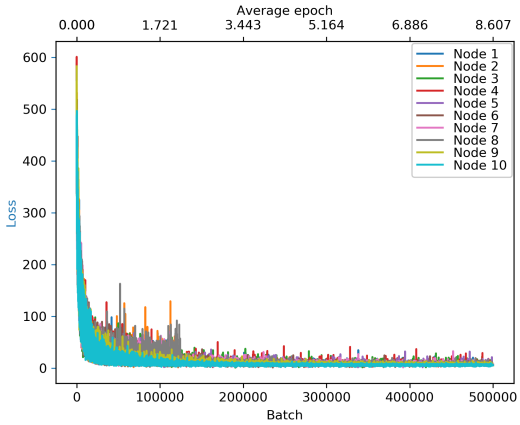

Figure 5.19: Gossip learning with infrequent exchange (*half*) results: $loss$ values over local batches.

Likewise in this configuration, the $w2v_{sim}$ values do not converge in this sub-configuration, as shown in Figure 5.20. The $w2v_{sim}$ values for the nodes converge to their final values between 400,000 and 500,000 local batches as well. However, the local values range wider now from 17 to 35. This is significantly lower than the $w2v_{sim}$ values seen for the other sub-configurations, as is the case for the previous configuration as well. Similarly, there does not seem to be strong correlations between the topic of node content and its final $w2v_{sim}$ value. It can be said that in this sub-configuration, with more infrequent exchange of models, there is a wider range for the embedding quality.

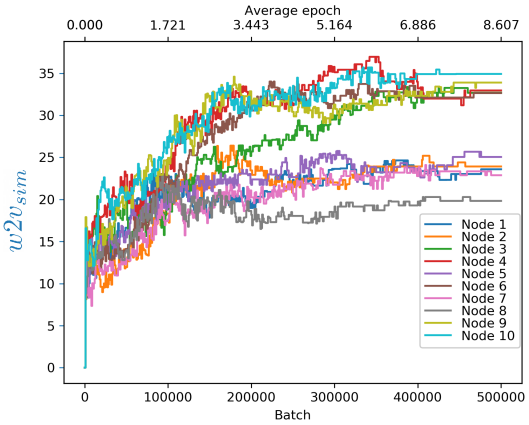

Figure 5.20: Gossip learning with infrequent exchange (*half*) results: $w2v_{sim}$ values over local batches. The final $w2v_{sim}$ values range between 17 and 35. In comparison, the baseline is 65.

Table 5.10 shows the target and corresponding predicted context words for the *half* sub-configuration for node 10, the node with the highest final $w2v_{sim}$ value. Again, the context words shown in the table are of lower quality semantically compared to the other sub-configurations, and more training iterations are not likely to improve this result as well as Figure 5.20 suggests.

The results of this configuration are summarized Table 5.11. The training time of all the sub-configurations are comparable to each other, as well as to the traditional centralized training configuration. But the training times are shorter than the experiments run using the gossip learning with frequent exchange configuration. In terms of Word2Vec similarity, the first three subcategories are comparable in their results, which seems to indicate (at least within the context of the configuration) that the heterogeneity and equality of sizes of the datasets are not a determining factor. The $w2v_{sim}$ values are slightly higher in this configuration as well, where in one case the saturation occurs before 400,000 batches while in some after. It is thus more likely that the $w2v_{sim}$ value will converge faster with more frequent exchange.

Table 5.10: Target words and predicted context words for gossip learning with infrequent exchange *half* at each local batch 500,000.

Target word	**Nearest predicted context words**
five	two, three, first, four, one, time, year, new
war	military, became, army, city, general, took, german, former
work	many, also, well, however, could, time, â^{TM}, made
year	later, following, years, new, began, since, second, two
people	according, many, however, among, order, said, could, work
location	another, period, across, times, UNK, one, called, two
october	december, march, january, april, september, july, august, june
state	government, states, national, members, according, people, general, political
science	women, well, â^{TM}, work, university, way, also, including
rights	public, said, us, would, since, national, world, including
history	work, early, according, one, well, called, among, made
money	reported, would, may, police, us, time, day, could
bank	years, day, new, became, since, including, made, group
man	one, time, also, since, world, however, made, early
woman	times, among, time, one, first, also, book, published
growth	considered, human, large, form, due, different, well, many
spring	last, became, history, general, began, church, german, war
life	many, however, like, work, could, well, within, also
hard	great, period, form, known, 18, large, major, society
culture	many, within, like, early, several, however, people, life
medium	possible, battle, side, started, forces, similar, meeting, german
family	species, found, known, UNK, south, near, area, north
alien	genocide, eggs, annual, week, 3, awards, music, moreover

Table 5.11: Summary of sub-configuration results of gossip learning with infrequent exchange.

Sub-configuration	Training time (h)	$w2v_{sim}$	Saturation batch
topicwise	19.403	62	320,000
randombalanced	19.734	61	430,000
randomimbalanced	21.454	61	430,000
half	19.373	17 - 35	400,000 - 500,000

5.4 Local Nodes Training with Common Vocabulary

In this configuration, each node trains locally while using the common vocabulary from all the nodes. Figure 5.21 shows the $loss$ values of the nodes over the local batches, which flatten off after 300,000 local batches for most nodes.

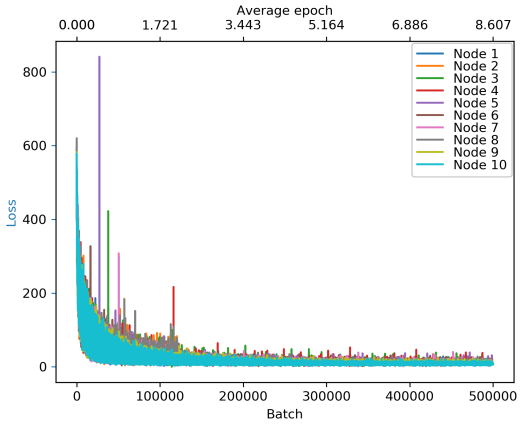

Figure 5.21: Local nodes training with common vocabulary results: $loss$ values over local batches.

The $w2v_{sim}$ values over the local batches are shown in Figure 5.22. The $w2v_{sim}$ values do not converge but this is to be expected as the models in each node trained independently. The $w2v_{sim}$ values for the nodes converge to their final values between 400,000 and 500,000 local batches, with the local values ranging from 39 to 57. There does not seem to be strong correlations between the topic of node content and its final $w2v_{sim}$ value as well.

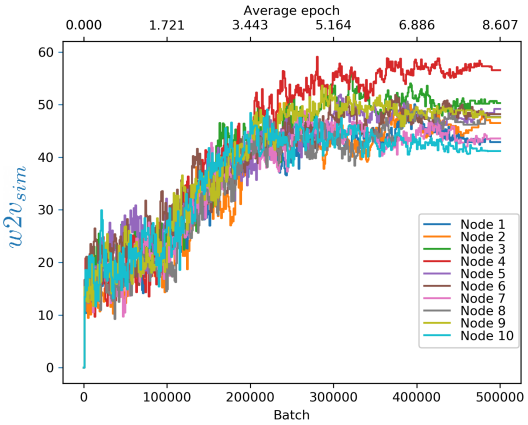

Figure 5.22: Local nodes training with common vocabulary results: $w2v_{sim}$ values over local batches. The final $w2v_{sim}$ values range between 39 and 57. In comparison, the baseline is 65.

What is more interesting to see however, is how the models predict the target words given their isolation and the homogeneity of the topic of the contents of their local datasets. Tables 5.12, 5.13, 5.14, 5.15, and 5.16 show the target words and their predicted context words for nodes 2, 4, 5, 8, and 9 respectively, whose respective datasets are homogeneously in the topic of *science*, *politics*, *business*, *humanities*, and *history*.

The initial hypothesis that each node would learn topic-specific words does not seem to be confirmed. For some of the nodes, the trained model are comparable to the other trained models in the other configurations. However, the results from this configuration solidifies the main purpose and needs of this project; within the constraints of the scenario of data privacy, the mere use of a common vocabulary without the exchange of models does not guarantee the convergence of the nodes in each model. Therefore, the gossip learning protocol implemented in this project does make sure that all nodes learn equally well.

Table 5.12: Target words and predicted context words for node 2 *science* for local nodes training with common vocabulary at batch 500,000.

Target word	**Nearest predicted context words**
five	three, four, six, eight, around, numerous, two, 25
war	military, army, leader, government, took, german, became, foreign
work	â^{TM}, would, working, idea, works, according, practice, created
year	years, first, 18, 17, 24, last, 2000, day
people	according, would, attempt, work, put, practice, idea, general
location	number, well, also, several, numbers, although, one, addition
october	april, september, november, december, march, july, june, february
state	led, world, local, established, rule, continued, came, according
science	scientific, work, research, working, society, works, community, association
rights	law, policy, legal, political, public, civil, laws, government
history	world, according, historical, modern, brought, led, works, popular
money	stated, would, issue, said, support, community, claims, working
bank	2006, 2005, 2000, 2004, established, capital, million, cities
man	never, according, wrote, woman, later, book, put, death
woman	man, never, according, go, attempt, mother, would, put
growth	low, produce, high, conditions, reduced, observed, types, higher
spring	summer, mostly, winter, mainly, near, sometimes, along, parts
life	others, way, even, could, fact, good, take, see
hard	often, like, much, unlike, form, may, either, inside
culture	history, literature, cultural, work, traditional, people, idea, influence
medium	low, easily, typical, reduced, contain, less, generally, depending
family	species, found, described, genus, endemic, america, southern, north
alien	crimes, serves, bias, transmission, plastic, column, appropriate, accurate

Table 5.13: Target words and predicted context words for node 4 *politics* for local nodes training with common vocabulary at batch 500,000.

Target word	**Nearest predicted context words**
five	three, four, two, first, years, six, place, included
war	military, german, british, joined, remained, declared, took, forces
work	many, various, â^{TM}, given, even, become, like, based
year	first, started, began, six, second, day, following, next
people	however, many, even, others, among, although, considered, take
location	large, small, main, across, parts, consists, around, smaller
october	july, april, june, september, november, march, august, december
state	country, states, members, independent, general, led, national, government
science	research, scientific, academic, study, students, focus, applied, work
rights	law, legal, freedom, justice, policy, political, laws, act
history	early, among, modern, historical, whose, according, though, believed
money	would, take, public, support, private, involved, could, us
bank	capital, 2000, local, office, established, largest, 1, year
man	whose, young, according, believed, described, though, wrote, woman
woman	man, men, young, life, described, upon, believed, though
growth	increase, increasing, increased, level, overall, conditions, less, higher
spring	along, around, two, three, end, part, known, main
life	even, way, others, fact, thus, good, considered, rather
hard	make, able, enough, without, much, way, similar, possible
culture	cultural, view, influence, idea, term, thought, traditional, rather
medium	industrial, water, existing, typically, length, structures, loss, structure
family	early, whose, death, according, called, came, prominent, great
alien	ruins, punjab, 1955, rapid, horses, error, suit, google

Table 5.14: Target words and predicted context words for node 5 *business* for local nodes training with common vocabulary at batch 500,000.

Target word	**Nearest predicted context words**
five	six, four, three, second, first, last, followed, third
war	military, german, army, germany, british, members, member, french
work	described, others, become, although, take, working, one, though
year	began, new, started, 12, 15, first, 25, 14
people	others, among, considered, according, believed, practice, life, thought
location	across, also, well, one, called, open, around, several
october	november, december, july, january, february, april, march, september
state	act, government, law, states, society, foreign, opposed, country
science	idea, concept, work, described, history, ideas, others, creation
rights	states, government, national, members, nations, act, article, civil
history	said, great, among, early, whose, according, brought, despite
money	interest, pay, account, private, financial, report, public, paid
bank	financial, capital, private, pay, companies, money, investment, banks
man	young, said, death, history, never, wrote, family, whose
woman	man, death, young, family, men, life, mother, children
growth	increase, increasing, increased, less, reduced, greater, loss, higher
spring	around, along, across, region, part, years, almost, end
life	others, thought, people, idea, considered, upon, described, believed
hard	similar, unlike, like, use, addition, using, either, allowing
culture	others, influence, idea, among, life, people, considered, believed
medium	complex, produce, enough, used, simple, usually, different, quality
family	history, whose, brought, young, said, great, old, death
alien	spirit, want, nazi, berlin, elections, 1945, job, conservative

Table 5.15: Target words and predicted context words for node 8 *humanities* for local nodes training with common vocabulary at batch 500,000.

Target word	**Nearest predicted context words**
five	three, four, six, around, eight, 12, two, almost
war	military, army, german, took, government, became, campaign, russian
work	â^{TM}, would, working, according, created, works, interest, practice
year	years, first, last, day, 24, 2000, late, ten
people	according, work, community, others, put, would, attempt, practice
location	well, move, also, number, like, one, shows, numbers
october	september, april, november, december, july, june, february, march
state	established, led, world, local, according, rule, association, members
science	scientific, work, community, working, society, research, interest, future
rights	legal, political, law, policy, public, organizations, organization, justice
history	historical, according, world, book, brought, led, wrote, established
money	would, put, working, stated, private, issue, business, offered
bank	2000, country, 2005, began, city, 2001, 2004, former
man	never, according, wrote, later, woman, death, said, came
woman	man, never, later, according, death, go, mother, gave
growth	produce, low, observed, high, conditions, reduced, types, presence
spring	summer, mostly, near, winter, growing, mainly, parts, sometimes
life	others, way, good, see, find, even, however, fact
hard	much, like, entire, completely, unlike, form, two, often
culture	cultural, people, history, literature, influence, according, scholars, idea
medium	present, generally, relatively, observed, single, typical, often, form
family	species, found, genus, described, endemic, southern, america, south
alien	serves, opportunity, relevant, committed, situations, vessels, concerned, questions

Table 5.16: Target words and predicted context words for node 9 *history* for local nodes training with common vocabulary at batch 500,000.

Target word	**Nearest predicted context words**
five	four, three, six, first, new, since, second, two
war	military, german, army, forces, declared, remained, british, germany
work	based, like, others, described, another, way, considered, even
year	new, 12, years, six, first, 30, four, 10
people	many, however, although, within, among, especially, particularly, become
location	around, main, along, large, across, parts, far, side
october	march, april, november, september, december, august, july, june
state	states, majority, foreign, independent, country, led, established, authorities
science	scientific, approach, concept, research, idea, study, theory, particular
rights	law, policy, act, political, legal, states, members, support
history	according, among, modern, early, whose, although, referred, influence
money	would, public, take, making, stated, get, involved, help
bank	national, country, local, regional, total, united, largest, approximately
man	wrote, said, described, come, book, believed, upon, instead
woman	young, man, said, never, upon, wrote, described, another
growth	increased, level, increase, increasing, rate, higher, low, due
spring	around, across, two, three, along, back, side, days
life	way, rather, others, even, thus, considered, means, form
hard	use, make, create, may, using, multiple, uses, used
culture	cultural, traditional, modern, important, history, considered, especially, various
medium	generation, metal, value, change, paper, quite, changes, hot
family	among, according, whose, later, known, early, although, made
alien	strategic, grow, clearly, heat, feet, conventional, signal

5.5 Discussion

For both of the configurations of gossip learning with frequent and infrequent exchange, the model quality for the first three sub-configurations are quite comparable to the traditional centralized configuration as per the $w2v_{sim}$ values. In fact, for the gossip learning with infrequent exchange configuration, there is a slight improvement over the infrequent exchange in terms of training time required and $w2v_{sim}$ value. This indicates that there is not a significant difference between the frequency of model sharing between both configurations despite the 50 times reduction in communications. Therefore, it would be interesting for future work to figure out the optimal hyperparameters $localsteps$ and $GLsteps$. Moreover, taking 65 as the midpoint of the $w2v_{sim}$ value for the traditional centralized training, using the implemented gossip algorithm for the fist three sub-configurations, there is between 4.725% and 8% decrease in the value, and that averages to 6.904%.

Furthermore, the relatively unchanged values of $w2v_{sim}$ for the first three sub-configurations in spite of the heterogeneity/homogeneity of the node contents and their size shows that the implemented gossip algorithm (at least under the parameters specified) is robust to topicality and local datasets size.

Sharing the M_2 matrix in the *half* sub-configurations does not yield a favorable result in both cases, which calls for further optimization techniques in order to reduce bandwidth. And training the models locally using common vocabulary does not guarantee the convergence of all the nodes. The models of each node in a network trained using gossip learning converge with similar quality.

Chapter 6

Conclusions and Future work

The conclusions from this project are presented in this chapter in Section 6.1. Furthermore, the limitations of this project and consequent potential future work are presented and suggested in Sections 6.2 and 6.3 respectively. Finally, Section 6.4 gives the benefits and ethical considerations of this project.

6.1 Conclusions

In line of the purpose of the project introduced in Section 1.3, this work has fulfilled the goals stated in Section 1.4 under the assumption also detailed in Section 3.1.4. Using the methodology and evaluation explained in Chapter 3, experiments following the configurations and sub-configurations described in Section 4.2 were carried out and their corresponding results and analysis are showed and discussed in Chapter 5. These results corroborate the main hypothesis of this project, i.e. the viability of massively-parallel, data-private, decentralized approach for an NLP algorithm.

Motivated by the scenario where various organizations wish to make use of the corpora from other organizations for the purpose of NLP training without disclosing their own sensitive data, the purpose of this project is to test whether a real-world application of Word2Vec running on massively-parallel, decentralized, data-private framework. The main contributions of this work are

- the implementation of Word2Vec algorithm using the gossip learning approach for large-scale, real-world applications;
- the identification of parameters and circumstances under which the gossip learning approach produces good enough results - comparable

to traditional centralized training;

- the evaluation, analysis and comparison of embedding quality trained using gossip learning with respect to two orthogonal aspects of node datasets: topic-distribution and size-distribution;
- the implementation of some technique of bandwidth saving and its effect on the embedding quality.

The following research question was introduced in Section 1.3

> ***How do models that are produced from the corpus of each node on a decentralized, fully-distributed, data-private configuration, i.e. gossip learning, compare to that trained using a traditional centralized approach where all the data are moved from the local machines or devices using comparable parameters with respect to several evaluations?***

With references to the results presented in Chapter 5, this project provides the following answer

> ***The quality of word embeddings produced using the gossip learning approach is comparable to that trained in a traditional centralized configuration using the same parameters, with an average loss on quality of 6.904%. The frequency of model exchange, which costs bandwidth, has also been reduced 50 times without loss of embedding quality. The results of this project therefore show that gossip learning is viable for large-scale, data-private NLP implementation for real-world applications.***

Overall, the conclusion of this project is that gossip learning can be applied to NLP scenarios where data privacy is a major concern. However, due to the novelty of the research area, this project has its limitations as further explained in Section 6.2.

6.2 Limitations

With the decentralized machine learning field largely left unexplored, particularly concerning the application of NLP, the explorations and experiments carried out in this project are chiefly to test out the implementation of Word2Vec on gossip learning with data privacy in consideration. Although the experimental setup of this project takes into account parameters and conditions which simulate real-world scenarios, it is still limited in some scopes, such as the network conditions.

The network conditions under which the experiments were run are assumed to be perfect. Therefore, nodes in the network never drop due to network issues - although this is effectively mitigated (as presented in other literature) by increasing the number of nodes in the network. The exchange of weight matrices of the nodes' models is therefore also considered virtually instantaneous, which explains the relatively similar training time for all the experiments (given the equal number of batches processed in each run).

Additional security and privacy considerations in the context of network were not taken into account when carrying out the experiments. Although they were never the focus of this research, their significance cannot be overlooked as they bolster the purpose of this project.

From the point of view of algorithms, only a single NLP algorithm and task is evaluated and compared. This is mainly due to the purpose of this research which focuses on testing the viability of gossip learning and comparing it to its centralized counterpart. Nonetheless, this project does not extend to other algorithms or NLP tasks.

Furthermore, the experiments were not performed with the goal of tuning the hyperparameters. However, even without extensive hyperparameter optimization, the results showed satisfactory performance of the gossip learning algorithm. Therefore, it cannot be ruled out that with proper hyperparameter optimization, the embedding quality when using the gossip learning approach can match or even surpass that from traditional centralized learning.

6.3 Future work

The field of decentralized being still in its fledgling state, and in particular with respect to NLP applications, there are still large areas unexplored. This project has shed some light on the possibility of deploying an NLP algorithm on gossip learning.

However, this means that there are still many potential scenarios that have not been considered in this research. Future research should explore all these scenarios beyond what was described in Section 4.2. Scenarios or configurations leading to useful results that can be the focus of future research include the evaluation and comparison of various NLP algorithms in different configurations using gossip learning. Moreover, formal comparisons under similar parameters with alternative massively-parallel, data-private approaches, such as federated learning, are another avenue to explore.

Furthermore, a technique to reduce the bandwidth cost for sending the weights between the peers by sending only one of the weight matrices has been implemented and evaluated. However, this yielded unfavorable results when it comes to the final quality of the embeddings. This calls for more research in this area. Decreasing the frequency of sharing only one matrix such that it happens only once in a while can be investigated, combined also with other techniques such as compression techniques on the vectors. The trade-off between embedding quality and bandwidth cost saving is an interesting question to investigate.

6.4 Reflections

One of the most important results of this project is the confirmation of the viability of massively-parallel, data-private configuration for text data in the form of corpora for NLP applications. As such, the findings of this research can hopefully be used as a measure to preserve privacy of data, and to show that a massively-parallel, data-private, decentralized protocol can give good enough results compared to traditional centralized learning as well as datacenter-scale configurations.

This project will hopefully also contribute towards the research in the decentralized machine learning field and spark more interest in further research. Therefore, future research may build upon the findings of presented in this project to expand the knowledge in the field, which arguably is still in its infancy.

References

[1] Y. LeCun, "Yoshua bengio, and geoffrey hinton," *Deep learning. nature*, vol. 521, no. 7553, pp. 436–444, 2015.

[2] E. P. Xing, Q. Ho, P. Xie, and D. Wei, "Strategies and principles of distributed machine learning on big data," *Engineering*, vol. 2, no. 2, pp. 179–195, 2016.

[3] Q. Yang, Y. Liu, T. Chen, and Y. Tong, "Federated machine learning: Concept and applications," *ACM Transactions on Intelligent Systems and Technology (TIST)*, vol. 10, no. 2, pp. 1–19, 2019.

[4] R. Ormándi, I. Hegedűs, and M. Jelasity, "Gossip learning with linear models on fully distributed data," *Concurrency and Computation: Practice and Experience*, vol. 25, no. 4, pp. 556–571, 2013.

[5] G. G. Chowdhury, "Natural language processing," *Annual review of information science and technology*, vol. 37, no. 1, pp. 51–89, 2003.

[6] T. Mikolov, I. Sutskever, K. Chen, G. S. Corrado, and J. Dean, "Distributed representations of words and phrases and their compositionality," in *Advances in neural information processing systems*, 2013, pp. 3111–3119.

[7] J. Deng, W. Dong, R. Socher, L.-J. Li, K. Li, and L. Fei-Fei, "Imagenet: A large-scale hierarchical image database," in *2009 IEEE conference on computer vision and pattern recognition*. Ieee, 2009, pp. 248–255.

[8] I. Krasin, T. Duerig, N. Alldrin, V. Ferrari, S. Abu-El-Haija, A. Kuznetsova, H. Rom, J. Uijlings, S. Popov, A. Veit *et al.*, "Openimages: A public dataset for large-scale multi-label and multi-class image classification," *Dataset available from https://github.com/openimages*, vol. 2, no. 3, p. 18, 2017.

[9] K. Fatahalian, J. Sugerman, and P. Hanrahan, "Understanding the efficiency of gpu algorithms for matrix-matrix multiplication," in *Proceedings of the ACM SIGGRAPH/EUROGRAPHICS conference on Graphics hardware*, 2004, pp. 133–137.

[10] K. Barkalov and V. Gergel, "Parallel global optimization on gpu," *Journal of Global Optimization*, vol. 66, no. 1, pp. 3–20, 2016.

[11] J. Verbraeken, M. Wolting, J. Katzy, J. Kloppenburg, T. Verbelen, and J. S. Rellermeyer, "A survey on distributed machine learning," *ACM Computing Surveys (CSUR)*, vol. 53, no. 2, pp. 1–33, 2020.

[12] J. Chen, X. Pan, R. Monga, S. Bengio, and R. Jozefowicz, "Revisiting distributed synchronous sgd," *arXiv preprint arXiv:1604.00981*, 2016.

[13] P. Voigt and A. Von dem Bussche, "The eu general data protection regulation (gdpr)," *A Practical Guide, 1st Ed., Cham: Springer International Publishing*, 2017.

[14] H. B. McMahan, E. Moore, D. Ramage, S. Hampson *et al.*, "Communication-efficient learning of deep networks from decentralized data," *arXiv preprint arXiv:1602.05629*, 2016.

[15] G. Ateniese, L. V. Mancini, A. Spognardi, A. Villani, D. Vitali, and G. Felici, "Hacking smart machines with smarter ones: How to extract meaningful data from machine learning classifiers," *International Journal of Security and Networks*, vol. 10, no. 3, pp. 137–150, 2015.

[16] Y.-A. De Montjoye, L. Radaelli, V. K. Singh *et al.*, "Unique in the shopping mall: On the reidentifiability of credit card metadata," *Science*, vol. 347, no. 6221, pp. 536–539, 2015.

[17] P. Vanhaesebrouck, A. Bellet, and M. Tommasi, "Decentralized collaborative learning of personalized models over networks," 2017.

[18] A. Soliman, S. Girdzijauskas, M.-R. Bouguelia, S. Pashami, and S. Nowaczyk, "Decentralized and adaptive k-means clustering for non-iid data using hyperloglog counters," in *Pacific-Asia Conference on Knowledge Discovery and Data Mining*. Springer, 2020, pp. 343–355.

[19] L. Giaretta and Š. Girdzijauskas, "Gossip learning: off the beaten path," in *2019 IEEE International Conference on Big Data (Big Data)*. IEEE, 2019, pp. 1117–1124.

[20] D. Kempe, A. Dobra, and J. Gehrke, "Gossip-based computation of aggregate information," in *44th Annual IEEE Symposium on Foundations of Computer Science, 2003. Proceedings.* IEEE, 2003, pp. 482–491.

[21] E. Wei and A. Ozdaglar, "Distributed alternating direction method of multipliers," in *2012 IEEE 51st IEEE Conference on Decision and Control (CDC)*. IEEE, 2012, pp. 5445–5450.

[22] D. Shah, "Network gossip algorithms," in *2009 IEEE International Conference on Acoustics, Speech and Signal Processing*. IEEE, 2009, pp. 3673–3676.

[23] A. Demers, D. Greene, C. Hauser, W. Irish, J. Larson, S. Shenker, H. Sturgis, D. Swinehart, and D. Terry, "Epidemic algorithms for replicated database maintenance," in *Proceedings of the sixth annual ACM Symposium on Principles of distributed computing*, 1987, pp. 1–12.

[24] R. Van Renesse, Y. Minsky, and M. Hayden, "A gossip-style failure detection service," in *Middleware'98*. Springer, 1998, pp. 55–70.

[25] Á. Berta and M. Jelasity, "Decentralized management of random walks over a mobile phone network," in *2017 25th Euromicro International Conference on Parallel, Distributed and Network-based Processing (PDP)*. IEEE, 2017, pp. 100–107.

[26] I. Hegedűs, Á. Berta, L. Kocsis, A. A. Benczúr, and M. Jelasity, "Robust decentralized low-rank matrix decomposition," *ACM Transactions on Intelligent Systems and Technology (TIST)*, vol. 7, no. 4, pp. 1–24, 2016.

[27] R. Collobert, J. Weston, L. Bottou, M. Karlen, K. Kavukcuoglu, and P. Kuksa, "Natural language processing (almost) from scratch," *Journal of machine learning research*, vol. 12, no. Aug, pp. 2493–2537, 2011.

[28] N. Kalchbrenner and P. Blunsom, "Recurrent continuous translation models," in *Proceedings of the 2013 Conference on Empirical Methods in Natural Language Processing*, 2013, pp. 1700–1709.

[29] D. Chen and C. D. Manning, "A fast and accurate dependency parser using neural networks," in *Proceedings of the 2014 conference on empirical methods in natural language processing (EMNLP)*, 2014, pp. 740–750.

[30] T. Mikolov, K. Chen, G. Corrado, and J. Dean, "Efficient estimation of word representations in vector space," *arXiv preprint arXiv:1301.3781*, 2013.

[31] "Google code archive - long-term storage for google code project hosting." [Online]. Available: https://code.google.com/archive/p/word2vec/

[32] O. Barkan and N. Koenigstein, "Item2vec: neural item embedding for collaborative filtering," in *2016 IEEE 26th International Workshop on Machine Learning for Signal Processing (MLSP)*. IEEE, 2016, pp. 1–6.

[33] Q. Le and T. Mikolov, "Distributed representations of sentences and documents," in *International conference on machine learning*, 2014, pp. 1188–1196.

[34] F. Morin and Y. Bengio, "Hierarchical probabilistic neural network language model." in *Aistats*, vol. 5. Citeseer, 2005, pp. 246–252.

[35] A. Mnih and Y. W. Teh, "A fast and simple algorithm for training neural probabilistic language models," *arXiv preprint arXiv:1206.6426*, 2012.

[36] [Online]. Available: https://dumps.wikimedia.org/backup-index.html

[37] Attardi, "attardi/wikiextractor," Mar 2020. [Online]. Available: https://github.com/attardi/wikiextractor

[38] "Petscan." [Online]. Available: https://petscan.wmflabs.org/

[39] B. Wang, A. Wang, F. Chen, Y. Wang, and C.-C. J. Kuo, "Evaluating word embedding models: methods and experimental results," *APSIPA Transactions on Signal and Information Processing*, vol. 8, 2019.

[40] 3Top, "3top/word2vec-api," Nov 2017. [Online]. Available: https://github.com/3Top/word2vec-api#where-to-get-a-pretrained-models

Appendix A

List of stop words

The list of stop words is [’a’, ’about’, ’above’, ’after’, ’again’, ’against’, ’ain’, ’all’, ’am’, ’an’, ’and’, ’any’, ’are’, ’aren’, "aren’t", ’as’, ’at’, ’be’, ’because’, ’been’, ’before’, ’being’, ’below’, ’between’, ’both’, ’but’, ’by’, ’can’, ’couldn’, "couldn’t", ’d’, ’did’, ’didn’, "didn’t", ’do’, ’does’, ’doesn’, "doesn’t", ’doing’, ’don’, "don’t", ’down’, ’during’, ’each’, ’few’, ’for’, ’from’, ’further’, ’had’, ’hadn’, "hadn’t", ’has’, ’hasn’, "hasn’t", ’have’, ’haven’, "haven’t", ’having’, ’he’, ’her’, ’here’, ’hers’, ’herself’, ’him’, ’himself’, ’his’, ’how’, ’i’, ’if’, ’in’, ’into’, ’is’, ’isn’, "isn’t", ’it’, "it’s", ’its’, ’itself’, ’just’, ’ll’, ’m’, ’ma’, ’me’, ’mightn’, "mightn’t", ’more’, ’most’, ’mustn’, "mustn’t", ’my’, ’myself’, ’needn’, "needn’t", ’no’, ’nor’, ’not’, ’now’, ’o’, ’of’, ’off’, ’on’, ’once’, ’only’, ’or’, ’other’, ’our’, ’ours’, ’ourselves’, ’out’, ’over’, ’own’, ’re’, ’s’, ’same’, ’shan’, "shan’t", ’she’, "she’s", ’should’, "should’ve", ’shouldn’, "shouldn’t", ’so’, ’some’, ’such’, ’t’, ’than’, ’that’, "that’ll", ’the’, ’their’, ’theirs’, ’them’, ’themselves’, ’then’, ’there’, ’these’, ’they’, ’this’, ’those’, ’through’, ’to’, ’too’, ’under’, ’until’, ’up’, ’ve’, ’very’, ’was’, ’wasn’, "wasn’t", ’we’, ’were’, ’weren’, "weren’t", ’what’, ’when’, ’where’, ’which’, ’while’, ’who’, ’whom’, ’why’, ’will’, ’with’, ’won’, "won’t", ’wouldn’, "wouldn’t", ’y’, ’you’, "you’d", "you’ll", "you’re", "you’ve", ’your’, ’yours’, ’yourself’, ’yourselves’]

For DIVA

{
"Author1": { "name": "Abdul Aziz Alkathiri"},
"Degree": {"Educational program": "Master's Programme, Computer Science, 120 credits"},
"Title": {
"Main title": "Decentralized Large-Scale Natural Language Processing Using Gossip Learning",
"Language": "eng" },
"Alternative title": {
"Main title": "Decentraliserad Storskalig Naturlig Språkbehandling med Hjälp av Skvallerinlärning",
"Language": "swe"
},
"Supervisor1": { "name": "Magnus Sahlgren (RISE SICS) Lodovico Giaretta (KTH)" },
"Examiner": {
"name": "Sarunas Girdzijauskas",
"organisation": {"L1": "School of Electrical Engineering and Computer Science" }
},
"Cooperation": { "Partner_name": "RISE SICS"},
"Other information": {
"Year": "2020", "Number of pages": "xiii,??"}
}

TRITA EECS-EX-2020:608