Comparative Analysis of SVM, BERT, and ERNIE in Relation Extraction Tasks

I Gusti Kresna
University of Manchester
igusti.kresna@postgrad.manchester.ac.uk

Kartiko Cokro Sewoyo
University of Manchester
kartikocokro.sewoyo@postgrad.manchester.ac.uk

Mohammad Bagus Pranata
University of Manchester
mohammadbagus.pranata@postgrad.manchester.ac.uk

Abstract

This study provides the implementation of Support Vector Machine (SVM) and Bidirectional Encoder Representations from Transformers (BERT) in the task of Relation Extraction (RE), followed by a comparative analysis between them and ERNIE (Zhang et al, 2019). Using the FewRel dataset, the models are evaluated based on precision, recall, and F1-score. BERT outperforms SVM, but ERNIE shows the highest performance in all metrics evaluation. The study underscores the importance of model selection based on specific task requirements and understanding the strengths and limitations of each model for an effective RE tasks.

1 Introduction

Relation Extraction is one of the key tasks of Natural Language Processing that identifies and classifies semantic relationships between entities in text. The relationships between these entities provide valuable information that can be used for various applications such as knowledge graph construction, question answering systems, and information retrieval (Bach & Badaskar, 2007).

Rule-based systems and supervised learning models, such as Support Vector Machines (SVM), are examples of traditional methods for RE (Joachims, 2002). Manual feature engineering is necessary for supervised learning models like SVM.

According to Devlin et al. (2018), transformer-based models such as BERT (Bidirectional Encoder Representations from Transformers) have demonstrated encouraging outcomes in a range of natural language processing tasks. BERT is an effective tool for RE because of its capacity to record contextual data in both left-to-right and right-to-left orientations (Wu & He, 2019).

This paper aims to explore and compare the performance of SVM and BERT that modelled on this research and the ERNIE from the previous work that use the same dataset.

2 Previous Work - ERNIE

Zhang et al. (2019) made a substantial addition to the field of RE by introducing ERNIE (Enhanced Representation through kNowledge IntEgration), a language representation model that is trained using Knowledge Graphs (KGs) and extensive textual datasets.

The ERNIE model is particularly relevant to the task of Relation Extraction (RE), as it leverages structured knowledge from KGs by including relations between entities. The authors demonstrate that ERNIE outperforms BERT on the RE task. This indicates the effectiveness of their approach in leveraging structured knowledge for relation extraction.

3 Methodology

3.1 Support Vector Machine (SVM)

3.1.1 SVM Method

SVM is a machine learning model used for classification by finding the hyperplane that best separates the data into different classes (Cortes & Vapnik, 1995). SVM can handle high dimensionality data well and therefore is largely used in various NLP tasks.

The initial SVM model is trained with a radial basis function (rbf) kernel and a default regularisation C. However, the best parameters are determined by the GridSearchCV that tests two different kernel functions and regularisation parameter C as shown on Appendix 1. The limited options are decided due to the limitation of computational resources.

3.1.2 SVM Features

In this study, some lexical, syntactic, and semantic features are extracted, which include Named Entity Recognition (NER), Part of Speech (POS) Tag, Dependencies, Word Distance, Word Embeddings, and TF-IDF as explained below. Moreover, the example of this feature is also shown on Appendix 2.

a. NER (Named Entity Recognition)

NER is extracted to determine the types of entities in head and tail, aiding in the understanding of the characteristics between each relation type.

b. Part of Speech (POS Tagging)

POS tag assigns labels to each word in a sentence according to its syntactic role. The POS tags for both the head and the tail are extracted according to their roles in the sentence and may aid in determining the relationship between the head and tail entities.

c. Dependency Relations

The dependency assigns a tree structure to a sentence, where nodes represent words and edges represent dependencies between words. This helps identify grammatical structure and word relationships for better relation type identification.

d. Distance

The distance is the number of words or tokens separating two entities, reflecting the syntactic structure and enabling the model to learn the typical distances for different types of relations.

e. Word Embeddings

Word embeddings are able to grasp the semantic relationships among words, aiding in the detection of distinct similarities among various relation labels. GloVe embeddings (Global Vectors for Word Representation) is used to capture the overall word-word co-occurrence patterns within a corpus.

f. Term Frequency - Inverse Document Frequency (TF-IDF)

The TF-IDF is applied to reflect how important a word is to a document and weigh the frequency since words may appear a lot of times but have little importance. The Term Frequency (TF) measures how frequently a word occurs in a document, while the Inverse Document Frequency (IDF) measures how important a word is.

The NER, POS Tags and Dependency are quantified by counting the number of unique values for head and tail present in each instance. For example, the POS tags of the example 39 are quantified into H_PROPN : 1 and T_PROPN = 1.

3.2 BERT

3.2.1 BERT Method

BERT, or Bidirectional Encoder Representations from Transformers, is a transformer-based model that uses bidirectional training to understand the context of a word based on all of its surroundings (left and right of the word) (Devlin et al., 2018). BERT can capture complicated semantic links within phrases and contextual nuances, it is favoured for Relation Extraction tasks as it improves the model's ability to identify complex associations between things.

Unlike the SVM method, there are no such ‘features’ that are used here. Instead, BERT automatically learns and extracts the datasets by using a form of word embedding, rather than being manually engineered as applied on SVM.

The BERT model in this relation extraction study is fine-tuned with a learning rate of 3e-5, 4 epochs, an epsilon value of 1e-8 for numerical stability, and a weight decay of 0.01 for regularisation.

3.2.2 BERT Preprocessing

Despite the automatic learning and extracting done by BERT, several manual settings are employed to get a better outcome as explained below. Moreover, the example of this preprocessing is also shown on Appendix 3 to help in predicting the relation.

a. Tokenization

The input sentences are tokenized using the BERT tokenizer by splitting the sentences into tokens, which can be words or subwords.

b. Special Tokens

Add special tokens ([E1], [/E1], [E2], [/E2]) to mark the start and end of head and tail entities within the sentences to improve the model's ability to recognize and differentiate between entities.

c. Padding

Ensure the pad sequences are shorter with special [PAD] tokens until they reach a specified maximum length of 128 tokens in order to guarantee that all sequences in a batch are the same length.

d. Truncation

If a sentence exceeds the maximum length of 128 tokens, it is truncated to fit within this limit.

e. Attention Mask Creation / Masking

Maskings are created to differentiate actual tokens from padding tokens. Using a binary vector, actual

tokens are marked with 1 and padding tokens are marked with 0.

3.3 Evaluation

Three metrics retrieval measures are widely used to assess a relation extractions model's efficacy on both the train and test sets: precision, recall, and F-score (Please refer to Appendix 4 of these metrics are measured).

4 Dataset

The dataset used in this study is the FewRel (Few-Shot Relation Classification Dataset) from Han et al., 2018. FewRel is a relation extraction dataset, which contains more than one hundred relations and tens of thousands of annotated instances.

For this study, the dataset is rearranged for the common classification settings, hence only using the training set from FewRel. The training set contains 64 relations with 100 human-annotated instances for each relation; the dataset is splitted into two sets, 80% for the training and 20% for the testing purpose with the detail as shown on Appendix 5.

5 Experimentation

5.1 SVM Experimentation

During the modelling, each of the feature combinations are tried to obtain the best result in predicting the test dataset. The combination of the features are displayed in Table 1.

Table 1. Feature Combinations

No.	Feature Combination
F1	NER + TF-IDF
F2	NER + TF-IDF + Distance + Word Embeddings
F3	NER + TF-IDF + Distance + Word Embeddings + Pos Tag
F4	NER + TF-IDF + Distance + Word Embeddings + Pos Tag + Dependency

These features then get scaled, since SVM is sensitive to the scale of the data, using the MaxAbsScaler before being used to train the SVM model. The MaxAbsScaler can deal with sparse data, since it scales the data by a constant factor and maintains the zero values.

During the modelling, each of the feature combinations are tried to obtain the best result in predicting the test dataset.

5.2 BERT Experimentation

Improving the BERT model is conducted to determine the best outcome in predicting the test dataset later. The details of modelling improvement and experimentation are explained below.

- *Tokenizer Customization*: the process of expanding the tokenizer's vocabulary with new unique tokens (['[E1]', '[/E1]', '[E2]', '[/E2]']) and resize the model's token embeddings for proper data comprehension.
- *Batch Size Adjustment*: reduce the batch size from four to two for better model performance and more updates, increasing iterations per epoch.
- Weight Decay Introduction in the Optimizer: use the weight decay (0.01) in the AdamW optimizer to minimise overfitting, resulting in a more generalised model by punishing big weights.
- *Enhanced Entity Position Handling*: sort the entity positions in reverse order before inserting special tokens to ensure correct placement.

6 Results & Discussion

6.1 SVM Results

Establishing the SVM model on the FewRel dataset involves extensive experimentation with various feature combinations, as outlined in Appendix 6, to yield optimal results.

The metrics performance indicates that the best model uses F4 feature combinations. It is also noted the biggest leap occurs with the addition of word embeddings as features, which provides unique semantic relation in each type.

In general, the SVM model can produce relatively good results in predicting the relation type, which is highly dependent on the feature extraction performed.

Additionally, the hyperparameter tuning confirms that the best hyperparameter of kernel function is the linear and the regularisation parameter C is one. The final outcomes of the predictions for the test set are displayed in Table 2, showcasing the top and bottom five results for each relation label.

Table 2. SVM final result after hyperparameter tuning.

Relation Label	P	R	F1
P1411	0.99	1	1
P105	0.99	0.99	0.99
P1435	0.99	0.99	0.99
P991	0.97	0.98	0.98
P1303	0.97	0.98	0.98
:	:	:	:
P551	0.58	0.6	0.59
P159	0.61	0.54	0.57
P58	0.58	0.49	0.53
P527	0.59	0.46	0.52
P127	0.46	0.48	0.47
macro avg	0.80	0.80	0.80
weighted avg	0.80	0.80	0.80

The low results mostly occur due to the similarity of the relation. For example, most errors in P127 are mistakenly labelled as P137, which are difficult to distinguish without further improvements in lexical resources and semantic understanding.

6.2 BERT Results

The performance of the BERT model on the test set achieves commendable results (Please refer to Appendix 7). The macro average of all metrics are at 0.86. This suggests that the BERT model generalizes well to unseen data and is effective at extracting relations from the text.

Moreover, Table 3 10 presents the score across the relations of the final results of the BERT model after improvements, sorted by F1 score. Each row represents a different relation label, with the corresponding metrics evaluation. It can be seen that the BERT performs exceptionally well on some relation labels similarly like on the SVM model (e.g., P105, P1435), achieving an F1 score close to 1. However, for some other labels (e.g., P127), the performance is relatively lower.

Table 3 BERT final result after model improvement

Relation Label	P	R	F1
P105	0.98	0.99	0.99
P1435	0.99	0.97	0.98
P1303	0.97	0.99	0.98
P118	0.97	0.99	0.98
P1344	0.98	0.95	0.97
:	:	:	:
P17	0.74	0.71	0.72
P58	0.76	0.64	0.70
P276	0.71	0.69	0.70
P131	0.69	0.69	0.69
P127	0.54	0.64	0.59
macro avg	0.87	0.86	0.86
weighted avg	0.87	0.86	0.86

6.3 Model Comparisons & Discussion

Appendix 8 makes clear that ERNIE has the best average performance when it comes to total performance across all metrics. However, the model performance's differences are not very noticeable compared to BERT or SVM.

In terms of language comprehension methods, ERNIE and BERT differ most from one another. BERT has a deep bidirectional understanding of language context, deriving word meanings from nearby text. ERNIE, on the other hand, expands on BERT's capabilities by including outside data from knowledge graphs. As a result, ERNIE is more effective at activities requiring knowledge because it can comprehend the context and interactions between texts better.

Furthermore, SVM and BERT can both handle the FewRel dataset's Relation Extraction task rather well. SVM exhibits a balanced performance with an F1-score of 0.8, recall, and precision of 0.8. BERT, on the other hand, performs better; its precision, recall, and F1-score are all 0.86. This implies that BERT performs better in accurately locating and categorising relationships within the dataset.

Whether the Relation Extraction task favours recall or precision, will determine which of these models is best. Furthermore, this model comparison implies that the particular requirements of the work at hand may influence the model selection. For example, ERNIE is the better option if high precision is needed. BERT, however, might be a good substitute if the task calls for a balance between recall and precision because of its similar F1-score.

7 Conclusion

This study concludes that BERT's transformer architecture outperforms SVM's extraction tasks. It is crucial to consider the specific details of a task when deciding whether to use BERT or ERNIE. While ERNIE has the advantage of incorporating external knowledge, the choice ultimately depends on whether recall, precision, or both are more important for the task. Therefore, it is crucial to thoroughly understand the strengths and weaknesses of each model in relation to the job at hand. Looking forward, investigating different models and feature combinations will be beneficial in further enhancing the performance of Relation Extraction tasks.

References

Jurafsky, D., & Martin, J. H. (2019). Speech and Language Processing (3rd ed.). [Online]. Available at: https://web.stanford.edu/~jurafsky/slp3/

Bach, N., & Badaskar, S. (2007, August). A review of relation extraction. Literature review for Language and Statistics II, 2(1).

Joachims, T. (2002). Learning to Classify Text Using Support Vector Machines (Vol. 668). Kluwer Academic Publishers.

Mintz, M., Bills, S., Snow, R., & Jurafsky, D. (2009, August). Distant supervision for relation extraction without labeled data. In Proceedings of the Joint Conference of the 47th Annual Meeting of the ACL and the 4th International Joint Conference on Natural Language Processing of the AFNLP: Volume 2-Volume 2 (pp. 1003-1011). Association for Computational Linguistics.

Devlin, J., Chang, M. W., Lee, K., & Toutanova, K. (2018). Bert: Pre-training of deep bidirectional transformers for language understanding. arXiv preprint arXiv:1810.04805.

Wu, Y., & He, W. (2019). Enriching pre-trained langclusteringuage model with entity information for relation classification. arXiv preprint arXiv:1905.08284.

Cortes, C., & Vapnik, V. (1995). Support-vector networks. Machine learning, 20(3), 273-297.

Vaswani, A., Shazeer, N., Parmar, N., Uszkoreit, J., Jones, L., Gomez, A. N., … & Polosukhin, I. (2017). Attention is all you need. In Advances in neural information processing systems (pp. 5998-6008).

Xu Han, Hao Zhu, Pengfei Yu, Ziyun Wang, Yuan Yao, Zhiyuan Liu, and Maosong Sun. 2018. FewRel: A large-scale supervised few-shot relation classification dataset with state-of-the-art evaluation. In Proceedings of the 2018 Conference on Empirical Methods in Natural Language Processing, pages 4803–4809, Brussels, Belgium. Association for Computational Linguistics.

Zhang, Z., Han, X., Liu, Z., Jiang, X., Sun, M., & Liu, Q. (2019). ERNIE: Enhanced Language Representation with Informative Entities. In Proceedings of the 57th Annual Meeting of the Association for Computational Linguistics (ACL), 1441-1451.

Zhao, J., Gui, T., Zhang, Q., & Zhou, Y. (2021). A Relation-Oriented Clustering Method for Open Relation Extraction. In Proceedings of the 2021 Conference on Empirical Methods in Natural Language Processing, pages 9707–9718

Appendix

Appendix 1. SVM Hyperparameter Tuning Value

No	Hyperparameter	Tuning Value
1	C Parameter	0.1, 1.0
2	Kernel	rbf, linear

Appendix 2. Example of Feature Extraction for SVM

Example 39: The Quad City Airshow is an annual event at the [Davenport Municipal Airport]$_{head}$ in [Davenport]$_{tail}$, Iowa and was the largest airshow in the state of Iowa .

Relation Label: P931
- Head: place served by transport hub
- Tail: territorial entity or entities served by this transport hub (airport, train station, etc.)

Feature	Feature Values
NER	head = {Davenport Municipal Airport: FAC}; tail = {Davenport: GPE}
POS Tags	head = {Davenport: PROPN, Municipal: PROPN, Airport: PROPN}; tail = {Davenport: PROPN}
Dependency	head = {Davenport: compound, Municipal: compound, Airport: pobj}; tail = {Davenport: pobj}
Distance	distance between head and tail = 1 word, [Davenport Municipal Airport]-- in --[Davenport]
Word Embeddings	GloVe 100 dimensional vectors of head and tail
TF-IDF	TF-IDF matrix of full sentence

Appendix 3. Example of BERT Preprocessing

Example 39: The Quad City Airshow is an annual event at the [Davenport Municipal Airport]$_{head}$ in [Davenport]$_{tail}$, Iowa and was the largest airshow in the state of Iowa .

Relation Label: P931
- Head: place served by transport hub
- Tail: territorial entity or entities served by this transport hub (airport, train station, etc.)

Feature	Feature Values
Tokenization	[“The”, “Quad”, “City”, “Airshow”, “is”, “an”, “annual”, “event”, “at”, “the”, “Davenport”, “Municipal”, “Airport”, “in”, “Davenport”, “,”, “Iowa”, “.”]
Special Tokens	[“The”, “Quad”, “City”, “Airshow”, “is”, “an”, “annual”, “event”, “at”, “the”, “[E1]”, “Davenport”, “Municipal”, “Airport”, “[/E1]”, “in”, “[E2]” “Davenport”, “[/E2]”“,”, “Iowa”, “.”]
Padding	[“The”, “Quad”, “City”, “Airshow”, “is”, “an”, “annual”, “event”, “at”, “the”, “[E1]”, “Davenport”, “Municipal”, “Airport”, “[/E1]”, “in”, “[E2]” “Davenport”, “[/E2]”“,”, “Iowa”, “.”, “[PAD]”,

	"[PAD]", …]
Truncation	If the sequence length exceeds the maximum length, truncate it (in this case, the model is set maximum length of 128 tokens)
Mask Creation	Indicate which tokens are real and which are padding: [1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 0, 0, …]

Appendix 4. Metric Evaluation for Relation Extraction Task

Metric	**Formula**
Accuracy	(TP + TN) / Total Population
Precision	TP / (TP + FP)
Recall	TP / (TP + FN)
F1-Score	(2 x Precision x Recall) / (Precision + Recall)

Definitions:

- True Positives (TP): Number of instances correctly predicted as positive.
- True Negatives (TN): Number of instances correctly predicted as negative.
- False Positives (FP): Number of instances incorrectly predicted as positive.
- False Negatives (FN): Number of instances incorrectly predicted as negative.

Appendix 5. Breakdown of the FewRel dataset being used

Training data		**Test data**	
Rows	Relation	Rows	Relation
35840	64	8960	64

Appendix 6. SVM performance through different feature combinations

Feature	**Dataset**	**P** (weighted avg)	**R** (weighted avg)	**F1** (weighted avg)
F1	Train	0.95	0.95	0.95
	Test	0.55	0.52	0.53
F2	Train	0.95	0.94	0.94
	Test	0.78	0.78	0.78
F3	Train	0.95	0.95	0.95
	Test	0.79	0.78	0.78
F4	Train	0.95	0.95	0.95
	Test	0.8	0.79	0.79

Appendix 7. BERT Train and Test set performance

Dataset	P (weighted avg)	R (weighted avg)	F1 (weighted avg)
Train	0.97	0.97	0.97
Test	0.86	0.86	0.86

Appendix 8. SVM, BERT, and ERNIE performance evaluation

Model	**Precision**	**Recall**	**F1-score**
SVM	0.80	0.80	0.80
BERT	0.86	0.86	0.86
ERNIE (Zhang et al, 2019)	0.88	0.88	0.88