Received: 4 January 2024 | Revised: 11 April 2024 | Accepted: 30 April 2024 | Published online: 30 May 2024

RESEARCH ARTICLE

Artificial Intelligence and Applications
2024, Vol. 2(4) 323–334
DOI: 10.47852/bonviewAIA42022424

BON VIEW PUBLISHING

Automatic Enemy Identification—Are We There Yet?

Huijuan Meng[1,*] and Jinshu Li[2]

[1]*Psychometrics and Data Science, Amazon Web Services, USA*

[2]*Exam Development, Amazon Web Services, USA*

Abstract: Enemy items refer to any two items that should not appear on the same test form. Accurately identifying enemy pairs is critical for ensuring the quality and fairness of exams, but it can also be challenging and time-consuming given the large number of possible item pairs in the exam item bank. Various enemy identification approaches have been explored to automate or semi-automate this task. In this process, the critical component is the encoding technique. The better the encoding technique captures the meaning of the sentences, the more accurate the similarity index and enemy classification results will be. This study focuses on evaluating the performance of a transformer-based model against the results from a string-based vector-space model (VSM) encoding technique under different research conditions for multiple-choice and multiple-response items used in a foundational information technology (IT) certification exam. The results suggest that when using sufficient representative training data and conducting fine-tuning, the transformer-based model significantly outperforms the VSM for enemy identification.

Keywords: enemy identification, text similarity, sentence transformer models, natural language processing, machine learning

1. Introduction

Enemy items refer to any two items that should not appear on the same test form [1]. Such items either measure similar content and are duplicative in nature [2], or one item provides information that could hint at the answer to another (cueing) [3]. Presenting enemy items on the same exam form can compromise measurement precision and jeopardize test validity. For testing programs with a large item bank, enemy identification becomes an ever-growing challenge for exam developers: every newly developed item must be compared against all existing items in the bank. Consequently, the number of item pairs requiring review increases exponentially, making manual review unfeasible.

Several methods have been proposed to identify enemy items and maintain the fairness and quality of exams. The process typically begins with an encoding technique to convert text items into a numerical format. This conversion is critical as it enables the calculation of similarity indices, which provide the basis for identifying enemy item pairs. Once the similarity indices have been computed, they can be utilized by either a machine learning (ML) algorithm or an arbitrary cutoff value to identify enemy item pairs. The use of ML algorithms is becoming increasingly popular due to their ability to account for complex relationships between item features. Finally, content experts review the flagged pairs to confirm their enemy status.

This study aims to evaluate and compare encoding techniques used for ML-based enemy identification of item pairs from a foundational IT certification exam item bank. It contrasts the traditional string-based vector-space model (VSM) with state-of-the-art transformer models.

*Corresponding author: Huijuan Meng, Psychometrics and Data Science, Amazon Web Services, USA. Email: huijuam@amazon.com

2. Literature Review

Text similarity is an important natural language processing (NLP) technique [4, 5] that enables machines to search through text corpora and identify relevant documents or passages [6]. Various numerical computations have been compared [7], and the accuracy of these indices heavily depends on the effectiveness of converting text to numbers. There are three main categories of encoding techniques used in NLP: string-based, knowledge-based, and corpus-based. Among these, corpus-based word embedding has emerged as the most popular and effective technique. This method represents each word in a text as a high-dimensional vector of numbers. Several methods have been proposed to generate word embeddings, including latent semantic analysis (LSA) [8], latent Dirichlet allocation (LDA) [9], Global Vectors for Word Representation (GloVe) [10], and Word2Vec [11]. More recently, transformer-based models [12] have revolutionized NLP research.

Transformer models such as BERT [13] and GPT [14] use a self-attention mechanism to encode the meaning of sentences or documents. Unlike other methods, transformer models can capture the context and dependencies among words in their embeddings, enabling more accurate and nuanced word representations that consider the full context in which a word appears.

However, for sentence similarity tasks, models developed particularly for this purpose, such as sentence transformer models [15], are preferable. These models modify the existing transformer network to derive semantically meaningful sentence embeddings, making them more effective for tasks like sentence similarity and semantic search. In contrast, general-purpose transformer models like BERT or RoBERTa may not be as effective for these tasks without additional fine-tuning [15].

© The Author(s) 2024. Published by BON VIEW PUBLISHING PTE. LTD. This is an open access article under the CC BY License (https://creativecommons.org/licenses/by/4.0/).

Despite the effectiveness of transformer-based text representation approaches, these methods have not been widely adopted in the field of educational measurement for enemy identification/classification research. Specifically, Fu and Han [16] conducted a study comparing the performance of 11 different text similarity indices for enemy pair classification, finding that cosine similarity obtained under the string-based VSM and the corpus-based LSA produced the best results for moderate corpus sizes (about 1,000 items/passages) with short to median document lengths (30–150 words per item/passage). Weir [1] improved classification accuracy and sensitivity in flagging enemy pairs by adding item bank metadata and subject matter experts' (SMEs) input to an LDA approach combined with ML classification techniques. Mao et al. [17] examined the effectiveness of using a string-based VSM-cosine similarity index (CSI) to flag enemy items in a medical licensure exam item bank. Peng [18] compared VSM, LDA, and LSA encoding techniques for enemy classification and found that VSM and LSA outperformed the LDA model. Finally, Becker and Kao [19] systematically conducted enemy identification across five item banks from different testing programs, computing several VSM-cosine similarities for the full item, question only, key only, question plus key, and non-key options. These CSI values, along with content area information, were used as predictors in a linear regression model to predict enemy relationships. Their work suggested that some cumulative distribution of item pairs across similarity categories can be helpful for determining a threshold for selecting item pairs for SMEs' review.

A common theme observed in these recent enemy identification studies is the exhaustive checking of all items in the bank/corpus, leading to the creation of thousands to millions of unique item pairs. These pairs have highly unbalanced distributions between enemy and non-enemy groups. All studies used bank enemy information for evaluating classification results, and SMEs reviewed some pairs and provided feedback, except for Fu and Han's [16] study.

Previous studies have made significant contributions to the field of automated enemy identification by shedding light on factors that may affect the results. However, these studies have not explored the use of transformer-based models, which are designed to produce semantically meaningful sentence embeddings and shown to outperform other models in benchmark NLP tasks such as text classification, question answering, and topic modeling [20, 21]. To address this gap, our study aims to investigate the effectiveness of sentence transformer models in the context of enemy identification.

3. Research Methodology

This section offers a detailed overview of the characteristics of the data and justifies the selection of transformer and ML models for classifying enemies. It compares five encoding techniques across ten diverse ML models. The workflow for enemy classification under various encoding conditions is graphically illustrated. Additionally, a strategic method is introduced to address data imbalance by leveraging item pair types and the stratified sampling technique. Finally, three training datasets are developed to investigate the impact of sample size and the representativeness of item pairs on the enemy classification results.

3.1. Item data

The data used in this study were derived from a foundational IT certification exam item bank, which comprised 1,649 multiple-choice items (with 4 options) and 305 multiple-response (MR) items (with 5 options). On average, each item contains approximately 30 words, including both the question and the answer key. The total of 1,954 items yields about 2 million unique item pairs.

This item bank has not undergone a thorough review by content experts specifically for enemy identification. It includes some known enemy pairs, with additional enemy relationships typically identified during the exam form review process. Consequently, only a small fraction of item pairs have a known status. Item pairs included in each published exam form are considered non-enemy pairs, as they have been manually reviewed by SMEs. Conversely, item pairs with existing enemy information are classified as enemy pairs. The status of the remaining pairs is unknown.

Table 1 summarizes the types of item pairs in this bank. Of the 1,908,081 unique item pairs, only 947 (0.05%) are labeled as "enemy." Furthermore, 27,433 pairs (1.44%) derived from individual exam forms are designated as "non-enemy." The remaining item pairs are divided into two groups based on their key comparison results: pairs where two items have at least 50% word overlap in the key are labeled as "uncertain with key overlap" (12,258, 0.64%); all other pairs are marked as "uncertain without key overlap" (1,867,441, 97.87%).

Key overlap is a strong indicator of enemy relationships: among enemy pairs, 69% exhibit key overlap, whereas for non-enemy pairs, this figure is only 0.39%. One method to quantify this feature is by computing the similarity between two keys [18, 19]. However, the item bank used in this study contains MR items, each with a key comprising two options, one of which may be used as the key for a multiple-choice question. Computing the similarity between the key of a multiple-choice item and the key of an MR item can yield misleading results. To address this issue, a key overlap variable was created to enhance the accuracy of classification results.

Furthermore, a string-based VSM-CSI was calculated for all item pairs in the bank. Results are summarized across five CSI intervals and four types of item pairs (Table 2). The distribution of CSI values reveals a clear pattern: most non-enemy item pairs have CSI values at or below 0.2, while a greater proportion of enemy item pairs have CSI values higher than 0.4. For uncertain item pairs without key overlap, the CSI distribution is similar to that of non-enemy pairs, although some pairs have higher CSI values. In contrast, uncertain item pairs with key overlap exhibit a unique CSI pattern: they tend to have a smaller proportion of high CSI values compared to enemy pairs and a smaller proportion of low CSI values compared to non-enemy pairs. CSI means were also compared across the four types of item pairs. The enemy group has the highest mean CSI value at 0.63, followed by the uncertain with key overlap group at 0.32. The mean CSI values for the non-enemy and uncertain without key overlap groups were around 0.05.

Table 1
Item pair distribution across different types

Item pair type	Pair *N*	Pair %	Pair with key overlap %
Enemy	947	0.05	69.13
Non-enemy (pairs within a form)	27,433	1.44	0.39
Uncertain with key overlap	12,258	0.64	100
Uncertain without key overlap	1,867,443	97.87	0
Total	1,908,081	100	0.68

Table 2
Item pair string-based VSM-CSI distribution across different types

CSI category	Non-enemy $N = 27,433$	Uncertain without key overlap: $N = 1,867,443$	Uncertain with key overlap: $N = 12,258$	Enemy $N = 949$
0.0~0.2	94.2	93.7	29.9	2
0.2~0.4	5.1	5.4	37.6	9.4
0.4~0.6	0.7	0.7	25.7	27.4
0.6~0.8	0	0.1	6.4	44.9
0.8~1.0	0	0	0.5	16.3
CSI Mean	0.049	0.051	0.319	0.63

Although the general trend suggests that higher text similarity often correlates with item pairs being classified as enemies, the CSI distributions indicate that text similarity alone might not be sufficient to determine an enemy relationship. Thus, other relevant information should also be considered when making this determination. This observation is consistent with enemy identification approaches in previous research [1, 18, 19].

3.2. Transformer model selection

The primary objective of this study is to assess the effectiveness of transformer-based models in the context of enemy identification. Two types of encoders can be used to generate similarity scores: bi-encoders and cross-encoders.

In a bi-encoder model, two sentences are encoded separately; the resulting sentence embeddings (two numerical vectors) are then used to compute a similarity index. In contrast, a cross-encoder model takes in a pair of sentences and encodes them together in a single encoding step, outputting a similarity score between 0 and 1. Figure 1 illustrates the workflow for the two types of encoders.

According to Reimers and Gurevych [15], cross-encoders achieve better performance than bi-encoders because they model interactions between sentences more explicitly. However, with the availability of more powerful bi-encoder models, such as MPNet [22–24], which was trained on large datasets, it is unclear whether the previous comparison results between cross-encoders and bi-encoders still hold true. Therefore, in this study, for each type of encoder, a well-performing model was selected and used to obtain similarity indices. Additionally, each model was run twice: once with additional training and once without. The latter is commonly referred to as the "out-of-the-box" model.

Table 3 summarizes the models evaluated in this study. For the bi-encoder, the "paraphrase-mpnet-base-v2" model was selected, which is trained on paraphrase datasets. The underlying transformer model is MPNet [22], which combines the strengths of masked and permuted language modeling for enhanced natural language understanding. MPNet has been shown to outperform BERT, XLNet, and RoBERTa models on multiple NLP benchmarks [23]. The "stsb-roberta-large" model has been selected for the cross-encoder due to its excellent performance on the semantic textual similarity (STS) benchmark [15]. STS is a widely used dataset for training and evaluating models on the task of measuring semantic similarity between sentences in a pair, which aligns with the goal of identifying enemy relationships between items. Finally, since the string-based VSM encoding technique was found to outperform other non-transformer techniques [16, 18], it was selected to compute the CSI, which is used to establish the evaluation baseline for enemy classification.

Figure 1
Bi-encoder and cross-encoder workflow

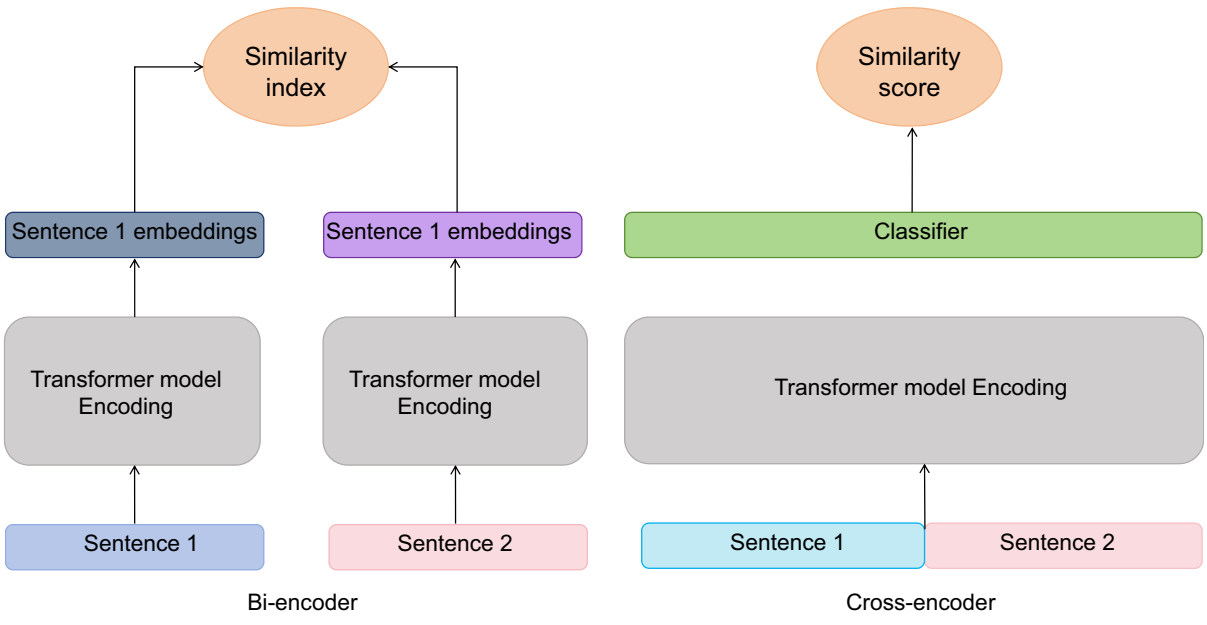


Table 3
Transformer models and baseline model

Encoding technique	Model
Bi-encoder	1. paraphrase-mpnet-base-v2: without training
	2. paraphrase-mpnet-base-v2: with training
Cross-encoder	3. stsb-roberta-large: without training
	4. stsb-roberta-large: with training
String-based (baseline)	5. Vector-space model

3.3. Machine-learning (ML) classification

Since the application of ML algorithms for classifying enemy pairs represents a novel approach in enemy identification research, it is uncertain which model might produce better results. Therefore, ten popular supervised ML models are trained to classify enemy pairs, including logistic regression (LR), linear discriminant analysis, quadratic discriminant analysis (QDA), *k*-nearest neighbors, naïve Bayes, support vector machine (SVM), decision tree (DT), random forest (RF), adaptive boosting, and neural networks. These models are selected since they vary in complexity and cover both linear and nonlinear approaches; technical details of these models can be found in Kuhn and Johnson [25].

Furthermore, this study focuses on comparing the performance of different encoding techniques in enemy identification. Therefore, default ML models are utilized without grid-search hyperparameter tuning. Conducting a grid search could result in varying ML model configurations for the examined encoding techniques, potentially confounding the comparison results. By maintaining constant ML hyperparameter values, any observed differences can be directly attributed to the encoding techniques themselves.

Four features are incorporated into the ML classification models, all of which demonstrate a certain degree of association with the enemy status of item pairs. These features and their respective correlation coefficients with enemy labels (calculated based on the data used in this study) are as follows:

1) Similarity index (0.66): This is a cosine similarity value from the baseline and bi-encoder models or a similarity score from the cross-encoder models. Here, the correlation is calculated based on the string-based VSM-CSI.
2) Key overlap (0.57): This indicates whether or not two items in the pair have at least 50% word overlap in the key.
3) Same topic (0.41): This indicates whether or not two items in the pair belong to the same content topic.
4) Same item type (0.10): This indicates whether or not two items in the pair have the same item type.

Finally, the classification label has two values: 0 indicates that the item pair is not an enemy, while 1 indicates that it is.

3.4. Enemy identification workflow

In this study, the enemy identification workflow is consistent with common practice, as summarized in the earlier section, except for the omission of the last step, which involves a review by SMEs. Figure 2 illustrates the process for enemy classification using encoding techniques without a training step, including the string-based VSM and two out-of-the-box sentence transformer models.

Essentially, item pairs from both the training and test data are input into the encoder to produce a similarity index for each pair. This value, along with three other features, namely, key overlap, same topic, and same item type, are used as predictors in the ML classification model. The ML model is trained with these predictors and the enemy labels from the training data, using default hyperparameter values. The trained ML model is then used to predict the enemy status of item pairs in the test data.

Figure 3 illustrates the encoding process involving training two sentence transformer models before using them to compute similarity values for each item pair. The only distinction from Figure 2 is that the sentence transformer models are first fine-tuned with the training data (item pairs and enemy labels). Then, the trained sentence transformer models encode item pairs from both the training and test datasets. After this step, the subsequent steps are identical to those in the encoding without the training process.

This training serves dual purposes: first, it enables the pretrained models to adapt to specific domain knowledge by exposing them to relevant content. Second, for cross-encoder models, training allows the model to adjust its weights to minimize the discrepancy between predicted and actual labels. In essence, the model learns to bring representations of enemy pairs closer together in the vector space while distancing those of non-enemy pairs.

In the sentence transformer training process, the batch size is set to 16, and the number of epochs is set to 1. This training could be configured with varying numbers of batches and epochs following a systematic comparison. However, this falls outside the scope of our current research. Considering the training dataset comprises at least 400 pairs, setting it to 16 batches—with 25 item pairs each—and one epoch is considered sufficient.

3.5. Enemy data structure

Item pairs in the bank are typically highly imbalanced between the enemy and non-enemy groups. Among all unique pairs in this bank, only 1.49% have a known enemy status, and these pairs exhibit very different text similarity patterns. Since enemy labels are required for training either the transformer model or ML model, or both, one approach to structuring the data is to use only pairs with a known status. Another approach is to manually add labels to some uncertain item pairs to increase the balance of the data. Given that most pairs in the bank fall under the unknown condition and need to be classified, the authors manually labeled an additional 600 item pairs, of which 400 were used in training and 200 in testing.

To examine the impact of data structure, two training datasets were constructed, each containing 800 item pairs. The first dataset is representative of the test data, where item pairs are relatively evenly distributed across all four types: enemy, non-enemy, uncertain with key overlap, and uncertain without key overlap. The second dataset contains only item pairs with known labels, half enemy and half non-enemy. Furthermore, to evaluate the impact of sample size on classification accuracy, a small training dataset was created using half of the representative training data.

Additionally, instead of randomly selecting items from each group, some strategies were implemented in creating training and test datasets. To ensure content coverage, for non-enemy item pairs selected from existing exam forms, each item is used in no more than two pairs. For the uncertain item pairs without key overlap, pairs with a VSM-CSI below 0.2 were excluded from selection to reduce the CSI mean value gap between this group and the uncertain item pairs with the key overlap group. Furthermore, stratified sampling was applied to ensure that both uncertain groups have similar CSI distributions.

More specifically, the mean CSI for the "uncertain with key overlap" group is 0.32, whereas the mean CSI values for the

Figure 2
Encoding without training workflow (vector-space model encoder and sentence transformer encoder)

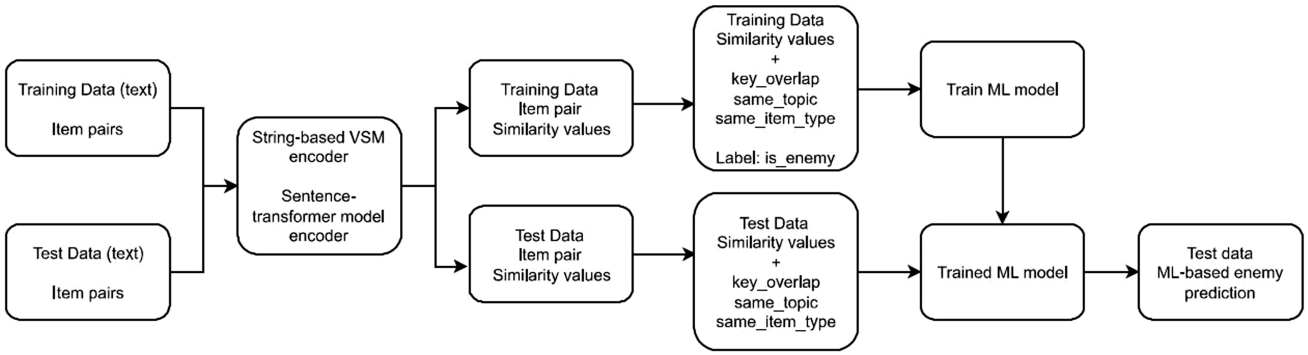
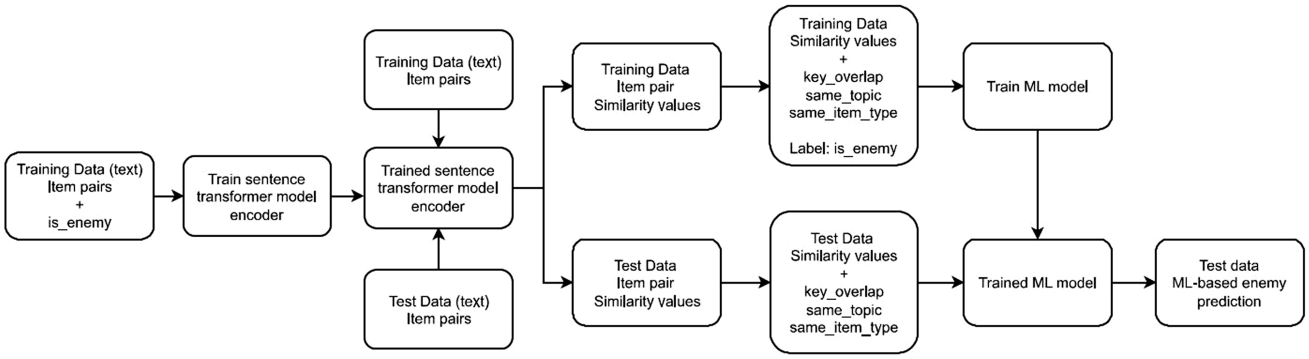


Figure 3
Encoding with training workflow (sentence transformer encoder)



"uncertain without key overlap" groups are around 0.05. To minimize the differences in mean CSI between these two groups, CSI intervals were established. By using the number of pairs within each interval from the "uncertain with key overlap" group as targets, an equivalent number of item pairs were selected from the "uncertain without key overlap" group. This stratified sampling approach ensures that the CSI means are comparable in both uncertain groups. It is presumed that this will increase the difficulty for transformer and ML models to learn the enemy relationship, potentially improving the classification accuracy of all pairs in the bank after training is complete.

In summary, four datasets were created as follows:

1) Large and representative training data: item pair $N = 800$
2) Large and non-representative training data: item pair $N = 800$
3) Small and representative training data: item pair $N = 400$
4) Test data: item pair $N = 400$

Table 4 summarizes the data structure of the three training datasets and the test dataset.

3.6. Evaluation criteria

Three evaluation metrics are computed for each model under each research condition, including accuracy, false positive rate (FP rate), false negative rate (FN rate). Accuracy is the proportion of correct ML classifications, but it can be misleading when evaluating results from imbalanced data. However, the data used in this study is specifically structured to be more or less balanced between two classification groups and across different types of item pairs. Therefore, evaluation metrics designed for imbalanced data such as the F1 score are not included.

There are other evaluation indices that can be used to assess ML classification performance, such as precision, recall, the receiver operating characteristic (ROC) curve, and the Kappa score. However, given that this study primarily evaluates the impact of different encoding techniques on enemy item classification, the chosen metrics (accuracy, FP rate, and FN rate) allow for a direct interpretation of the results. This approach enables readers to easily understand the differences—for instance, the false positive rate represents the percentage of items incorrectly classified as enemies. Utilizing Kappa scores or ROC curves could potentially complicate the interpretation of the outcomes, namely, the differences across different encoding techniques.

4. Results

In this section, we first compare the performance across five encoding approaches: string-based VSM, bi-encoder with and without training, and cross-encoder with and without training. Overall, the bi-encoder with domain knowledge adaptation training outperforms the other approaches, achieving the highest accuracy and the lowest rates of both false positives and false negatives.

We also examine the results across four item pair types: enemy, non-enemy, uncertain with key overlap, and uncertain without key

Table 4
Item pair data structure

Large and representative training data (item pair $N = 800$)							
Item pair type	Item pair N	Key overlap N	Same topic N	Same item type N	Enemy N	CSI mean	CSI SD
1. Non-enemy	200	2	86	146	0	0.10	0.14
2. Uncertain without key overlap	163	0	40	115	6	0.30	0.11
3. Uncertain with key overlap	237	237	186	174	126	0.31	0.11
4. Enemy	200	144	174	150	200	0.63	0.17
Large and non-representative training data (item pair $N = 800$)							
Item pair type	Item pair N	Key overlap N	Same topic N	Same item type N	Enemy N	CSI mean	CSI SD
1. Non-enemy	400	3	189	294	0	0.09	0.12
4. Enemy	400	281	358	304	400	0.63	0.18
Small and representative training data (item pair $N = 400$)							
Item pair type	Item pair N	Key overlap N	Same topic N	Same item type N	Enemy N	CSI mean	CSI SD
1. Non-enemy	100	1	40	76	0	0.11	0.13
2. Uncertain without key overlap	84	0	30	51	5	0.32	0.12
3. Uncertain with key overlap	116	116	92	82	61	0.30	0.09
4. Enemy	100	72	91	77	100	0.63	0.17
Test data (item pair $N = 400$)							
Item pair type	Item pair N	Key overlap N	Same topic N	Same item type N	Enemy N	CSI mean	CSI SD
1. Non-enemy	100	0	56	69	0	0.08	0.10
2. Uncertain without key overlap	91	0	25	74	3	0.28	0.09
3. Uncertain with key overlap	109	109	83	89	63	0.30	0.10
4. Enemy	100	66	85	86	100	0.63	0.19

overlap. These categories represent the structure of the item pairs studied here. The patterns in the results are consistent across various encoding techniques: compared to other item pair types, classification accuracy rates are significantly lower for item pairs in the "uncertain with key overlap" group. In this group, the best-performing approach, the bi-encoder with training, achieves only an 84.5% accuracy rate.

Additionally, the results are evaluated across ten supervised ML models. Simpler models, such as the QDA model and LR, yield more accurate results than complex models like the RF and DT.

Regarding the impact of training sample size (400 vs. 800) on enemy classification, using more item pairs with known status in the training dataset does not improve the outcome, except for the bi-encoder with training encoding approach.

Finally, the structure of the training data plays a crucial role in achieving better results. Among all unique item pairs in the item bank, less than 2% have a known enemy status; we are uncertain about the status of other pairs. To assist models in better encoding all types of item pairs, we intentionally constructed the test data to have a relatively equal distribution across item pair types. When the training data does not accurately represent the item pair types contained in the test data, the results are less satisfactory, even with a larger sample size.

4.1. Overall performance

In this study, we compare enemy classification results across five encoding approaches. The classification results, averaged across ten ML models, are summarized in Table 5. Without any domain knowledge adaptation training, both the bi-encoder and the cross-encoder models slightly underperformed compared with the string-based VSM, which served as the baseline encoding technique. The bi-encoder achieved an accuracy rate of 86.5%, slightly higher than the cross-encoder's rate of 86.3% by 0.2%, while the VSM reached an accuracy rate of 86.7%. However, after training with domain knowledge, the accuracy of the cross-encoder increased to 88.8%, and that of the bi-encoder rose to 93%. The best-performing sentence transformer model, the bi-encoder with training, outperformed the baseline model by 6.3%. Additionally, it achieved the lowest false positive rate (5.1%) and the lowest false negative rate (2%), representing improvements over the baseline approach by 2.9% and 3.3%, respectively.

4.2. Performance across item pair types

Results are further broken down by item pair types and summarized in Table 6. Please note that in this table and subsequent tables, the transformer-based encoding techniques are arranged in descending order based on their overall performance presented in Table 5, starting with bi-encoder without training, followed by cross-encoder without training, cross-encoder with training, and bi-encoder with training.

The largest performance differences are observed in the "uncertain with key overlap" group. With domain knowledge adaptation training, the bi-encoder achieved an accuracy of 84.5%, surpassing the baseline by 21.7% (62.8%). It also demonstrated better performance for item pairs classified as "enemy," albeit with a smaller margin (94.4% vs. 91.4%). However, for the "non-enemy" group and the "uncertain without key overlap" group, the baseline approach slightly outperformed the best transformer models by 0.5% and 1.1%, respectively. Notably, the patterns in performance differences for false positive and false negative rates mirror those observed in the accuracy results.

The cross-encoder, after domain knowledge adaptation training, achieved perfect accuracy (100%) in classifying item

Table 5
Overall[1] enemy identification results for different encoding techniques

Encoding technique	Accuracy rate %	FP rate %	FN rate %	Performance Order
String-based vector-space model (VSM)	86.7	8	5.3	3
Bi-encoder: without training	86.5	8.7	4.8	4
Bi-encoder: with training	93	5.1	2	1
Cross-encoder: without training	86.3	8.4	5.3	5
Cross-encoder: with training	88.8	7.6	3.5	2

[1]Results are averaged over ten ML classification outcomes

Table 6
Overall enemy identification results across encoding techniques and item pair type

	Accuracy rate %			
	Item pair type			
Encoding technique	Non-enemy	Uncertain without key overlap	Uncertain with key overlap	Enemy
String-based vector-space model (VSM)	99.3	96.2	62.8	91.4
Bi-encoder: without training	99.2	96.3	65.4	88
Cross-encoder: without training	99.7	94.1	66.6	87.2
Cross-encoder: with training	100	91.8	72.7	92.7
Bi-encoder: with training	98.8	95.1	84.5	94.4
	False positive rate %			
	Item pair type			
Encoding technique	Non-enemy	Uncertain without key overlap	Uncertain with key overlap	Enemy
String-based vector-space model (VSM)	0.7	2.9	25.1	1.2
Bi-encoder: without training	0.8	2.6	28.2	1
Cross-encoder: without training	0.3	4.8	26	0.6
Cross-encoder: with training	0	8	19.4	2
Bi-encoder: with training	1.2	3.8	13.3	1.2
	False negative rate %			
	Item pair type			
Encoding technique	Non-enemy	Uncertain without key overlap	Uncertain with key overlap	Enemy
String-based vector-space model (VSM)	0	1	12	7.4
Bi-encoder: without training	0	1.1	6.4	11
Cross-encoder: without training	0	1.1	7.4	12.2
Cross-encoder: with training	0	0.2	8	5.3
Bi-encoder: with training	0	1.1	2.2	4.4

pairs within the "non-enemy" group and outperformed the baseline approach for "enemy" and "uncertain with key overlap" groups. However, for item pairs in the "uncertain without key overlap" group, it produced the lowest accuracy rate (91.8%), even lower than that of the untrained cross-encoder approach. An examination of misclassified item pairs in this group revealed that the classification accuracy was adversely affected by an unusually high false positive rate.

Table 7 summarizes the baseline and transformer model's CSI values across false positive categories for the "uncertain without key overlap" group. Providing enemy labels to the default cross-encoder model during training resulted in an overemphasis on the textual differences between "enemy" and "non-enemy" item pairs. This led to high similarity scores for item pairs with higher VSM-CSI values and low similarity scores for those with lower VSM-CSI values. While this overemphasis proved effective for the "enemy" and "uncertain with key overlap" groups, where a "higher CSI indicates a higher likelihood of being an enemy" relationship holds true, it resulted in a significantly higher false positive rate for item pairs in the "uncertain without key overlap" group.

4.3. Performance across ML models

The ML classification results, averaged across five encoding techniques, are reported in Table 8. The quadratic discriminant analysis (QDA) model and LR both achieved the highest classification accuracy at 89.4%, surpassing the lowest performer, the DT model, by 3.6%. While the difference in the false positive rate between the highest and lowest performers is relatively small (7.1% vs. 8%), there is a wider gap in the false negative rate (2.6% vs. 7.1%).

Table 9 presents the best-performing ML model for each encoding technique. With the CSI outcome from the trained bi-encoder model, LR achieved the highest accuracy rate of 95%, with a false positive rate of 3.8% and a false negative rate of 1.2%. In comparison, for the baseline approach using the

Table 7
Baseline CSI and transformer CSI between false positive category (uncertain without key overlap group)

Encoding	False positive	VSM-CSI mean	Transformer-CSI mean
String-based vector-space model (VSM)	No	0.27	NA
	Yes	0.57	NA
Bi-encoder: without training	No	0.28	0.59
	Yes	0.46	0.78
Cross-encoder: without training	No	0.28	0.50
	Yes	0.37	0.68
Cross-encoder: with training	No	0.27	0.04
	Yes	0.44	0.77
Bi-encoder: with training	No	0.27	0.19
	Yes	0.50	0.60

Table 8
Overall[2] enemy identification results for different ML models

ML model	Accuracy rate %	FP rate %	FN rate %
Decision tree (DT)	85.8	7.1	7.1
Random forest (RF)	86.5	7	6.6
Support vector machine (SVM)	87.8	9.7	2.5
k-nearest neighbors (KNN)	88	7.3	4.7
Adaptive boosting (AdaBoost)	88.6	6.2	5.1
Linear discriminant analysis	88.8	8.5	2.6
Neural networks (NN)	89	7.1	3.9
Naïve Bayes (NB)	89.1	7.8	3.1
Logistic regression (LR)	89.4	6.9	3.7
Quadratic discriminant analysis (QDA)	89.4	8	2.6

[2]Results are averaged over five encoding technique outcomes

string-based VSM, accuracy dropped by 7%, the false positive rate increased by 5%, and the false negative rate increased by 2%.

The best ML results were further analyzed by item pair type and summarized in Table 10. Using the best ML model, the trained bi-encoder outperformed the baseline model on all evaluation metrics across all item types, except for the non-enemy group, where the performance was the same.

4.4. Impact of training data sample size on enemy identification results

Table 11 summarizes the performance differences resulting from varying the size of the training data. The large dataset contains 800 pairs, and the small one contains 400 pairs; both are relatively evenly distributed across four groups: enemy, non-enemy, uncertain without key overlap, and uncertain with key overlap. Among the five encoding techniques, the most significant difference was observed in the trained bi-encoder's results, where doubling the sample size increased the accuracy rate by 4.3% and reduced the FP rate by 1.8% and the FN rate by 2.4%. However, for the other techniques, the gains were minimal, and in two instances, the trained cross-encoder even achieved slightly better results with the smaller training sample.

Table 12 shows the performance differences across four item pair groups. For the trained bi-encoder, the size of the training data significantly affected enemy identification results for item pairs in the "uncertain with key overlap" group: with the smaller sample, accuracy dropped by 10.3%, the FP rate increased by 6.9%, and the FN rate increased by 3.4%. For item pairs in the "enemy" group, reducing the training dataset by half lowered the accuracy by 5.9% and increased the FN rate by 6.1%. For the other encoding techniques, the impact of varying the training data size on accuracy, FP rate, and FN rate was not consistent, and the differences in performance were relatively minor.

4.5. Impact of training data structure on enemy identification results

In this study, one training dataset was intentionally constructed to contain only known enemy and non-enemy item pairs. This data structure significantly and consistently affects enemy identification results, as shown in Table 13. With a non-representative training set, accuracy drops, ranging from 3.1 to 8.3% across encoding techniques, and the FP rates increased by 6.3% to 9.2%. However, the FN rates decreased, ranging from 1% for the trained bi-encoder to 4.3% for the baseline VSM approach. This means that with non-representative training data, more enemy pairs have been misclassified as non-enemies.

The results of the item pair type breakdown demonstrate similar patterns (Table 14), indicating that the training power significantly diminishes without a representative dataset. This effect is particularly pronounced for item pairs in the "uncertain with key overlap" group, where the accuracy can drop by up to 24.6%, and the FP rate can increase by up to 26.4 percentage points. However, this misalignment in data structure reduces FN rates for the "enemy" and "uncertain with key overlap" groups. Essentially, when the feature values in the training set significantly differ between the two groups being classified, such as the VSM-CSI mean being 0.08 for the "non-enemy" group and 0.63 for the "enemy" group, the similarity index may be overweighted in the classification, leading to more item pairs being classified as enemies. This results in higher FP rates and lower FN rates. Admittedly, the impact of training data structure on enemy classification results can be confounded by the training in the ML modeling, and further investigation is needed to disentangle the factors contributing to these results.

5. Discussions and Recommendations

In summary, this paper evaluates the performance of increasingly popular transformer-based models in the context of enemy identification, a topic not previously explored in this research area. Based on a literature review of non-transformer encoding techniques, the string-based VSM was chosen as the baseline and compared against four transformer-based approaches, including bi-encoder and cross-encoder, both with and without

Table 9
Best enemy identification results for each encoding technique

Encoding technique	ML model	Accuracy Rate %	FP rate %	FN rate %
String-based vector-space model	QDA	88	8.8	3.2
Bi-encoder: without training	LR	88.2	7.2	4.5
Cross-encoder: without training	AdaBoost	88.2	6.5	5.2
Cross-encoder: with training	QDA	89.8	8.2	2
Bi-encoder: with training	LR	95	3.8	1.2

Table 10
Best ML enemy identification results across encoding techniques and item pair type

Accuracy rate %				
	Item pair type			
Encoding technique	Non-enemy	Uncertain without key overlap	Uncertain with key overlap	Enemy
String-based vector-space model (VSM)	99	94.5	67	94
Bi-encoder: without training	100	98.9	70.6	86
Cross-encoder: without training	100	95.6	71.6	88
Cross-encoder: with training	100	91.2	73.4	96
Bi-encoder: with training	99	96.7	88.1	97
False positive rate %				
	Item pair type			
Encoding technique	Non-enemy	Uncertain without key overlap	Uncertain with key overlap	Enemy
String-based vector-space model (VSM)	1	4.4	26.6	1
Bi-encoder: without training	0	0	25.7	1
Cross-encoder: without training	0	3.3	21.1	0
Cross-encoder: with training	0	8.8	21.1	2
Bi-encoder: with training	1	2.2	11	0
False negative rate %				
	Item pair type			
Encoding technique	Non-enemy	Uncertain without key overlap	Uncertain with key overlap	Enemy
String-based vector-space model (VSM)	0	1.1	6.4	5
Bi-encoder: without training	0	1.1	3.7	13
Cross-encoder: without training	0	1.1	7.3	12
Cross-encoder: with training	0	0	5.5	2
Bi-encoder: with training	0	1.1	0.9	3

Table 11
Overall performance difference between large and small training datasets (large–small)

Encoding technique	Accuracy rate %	FP rate %	FN rate %
String-based vector-space model	0.5	−0.7	0.2
Bi-encoder: without training	0.7	−0.5	−0.2
Cross-encoder: without training	0.2	−0.1	−0.2
Cross-encoder: with training	−0.1	1.5	−1.5
Bi-encoder: with training	4.3	−1.8	−2.4

domain knowledge adaptation training. The two sentence transformer models evaluated in this study are "paraphrase-mpnet-base-v2" and "stsb-roberta-large." These models were pretrained with data for NLP tasks such as measuring semantic similarity between sentence pairs, which is similar to the enemy identification task. Given their superior performance across multiple NLP benchmarks, they were selected for this study. The similarity index values derived from these encoding techniques, along with three dichotomous predictors—key overlap, same topic, and same item type—were input into ten ML models to predict the enemy status for item pairs in the test data. Furthermore, three training datasets were created to evaluate the impact of sample size and data representativeness on enemy classification results.

Comparison results for encoding techniques revealed that, without domain knowledge adaptation training, transformer-based models did not produce better text representations, and thus, better enemy identification results than the non-transformer model. However, with training, the transformer models adapted to the domain knowledge and improved enemy classification accuracy by up to 6.3%. Among item pair groups, the best transformer-based model, the bi-encoder with training, outperformed the baseline model by 21.7% in accuracy. Another transformer-based

Table 12
Overall performance difference between large and small training datasets across encoding techniques and item pair type (large–small)

Accuracy rate %				
	Item pair type			
Encoding technique	Non-enemy	Uncertain without key overlap	Uncertain with key overlap	Enemy
String-based vector-space model (VSM)	−0.1	0.2	1.2	0.3
Bi-encoder: without training	−0.1	0.9	0.9	1.1
Cross-encoder: without training	0.5	0	0.2	0.2
Cross-encoder: with training	0.7	−3	−2	4.1
Bi-encoder: with training	0.3	−0.3	10.3	5.9
False positive rate %				
	Item pair type			
Encoding technique	Non-enemy	Uncertain without key overlap	Uncertain with key overlap	Enemy
String-based vector-space model (VSM)	0.1	0	−2.6	−0.1
Bi-encoder: without training	0.1	−0.9	−1.2	0.1
Cross-encoder: without training	−0.5	0	−0.1	0.4
Cross-encoder: with training	−0.7	3.7	2.8	0.2
Bi-encoder: with training	−0.3	0.3	−6.9	0.2
False negative rate %				
	Item pair type			
Encoding technique	Non-enemy	Uncertain without key overlap	Uncertain with key overlap	Enemy
String-based vector-space model (VSM)	0	−0.1	1.3	−0.2
Bi-encoder: without training	0	0	0.3	−1.2
Cross-encoder: without training	0	0	0	−0.6
Cross-encoder: with training	0	−0.7	−0.7	−4.3
Bi-encoder: with training	0	0	−3.4	−6.1

Table 13
Overall performance difference between representative and non-representative training datasets

Encoding technique	Accuracy rate %	FP rate %	FN rate %
String-based vector-space model	3.9	−8.2	4.3
Bi-encoder: without training	3.1	−6.3	3.2
Cross-encoder: without training	3.1	−6.8	3.6
Cross-encoder: with training	5.7	−7.9	2.1
Bi-encoder: with training	8.3	−9.2	1

model, the cross-encoder, yielded slightly better results but with an unusually high false positive rate for item pairs in the "uncertain without key overlap" group. Further exploration of item pairs under the false positive categories suggested that while cross-encoders could potentially perform better than bi-encoders by explicitly modeling interactions between sentences, they might overemphasize the relationship between text similarity and item pair enemy status, resulting in an inappropriately high similarity score. This overemphasis results in a higher false positive rate for items in the 'uncertain without key overlap' group, where only a few pairs are enemies, yet more pairs have higher CSI values—an outcome of the sample selection strategy utilized in this study. After learning about the enemy labels and item pairs during training, the cross-encoder model significantly increased the similarity scores for items with higher CSI values, leading to their incorrect identification as enemies in ML classification.

Ten ML models were evaluated across different encoding techniques for their classification accuracy. In general, the QDA and LR models produced the most accurate results, while the DT and RF models performed the worst. With one continuous and three dichotomous predictors, simpler ML models were more adept at determining the optimal weights for each feature and their interrelationships. Moreover, when the best ML model was used, the bi-encoder with training outperformed the baseline model on all evaluation metrics for all item types except for the non-enemy group, where its performance was tied to another encoding approach. It is noteworthy that this study employed default settings in ML classification to avoid confounding the results of the encoding technique comparison. However, systematic hyperparameter tuning could potentially improve the accuracy of enemy classification.

The impact of training dataset size and structure on enemy classification was also evaluated. When the training and test data contained a similar distribution of item pairs across the four groups (enemy, non-enemy, uncertain without key overlap, and uncertain with key overlap), halving the training data sample size did not significantly impact the enemy identification results, except for the best transformer-based approach, the bi-encoder with domain knowledge adaptation training. However, if the item pairs in the training dataset do not representatively match those in the test dataset, it can significantly lower classification accuracy across all encoding techniques. Based on this study's findings, it is recommended to train transformer-based encoding techniques using domain-specific data to enhance their text representation capability. Additionally, ensuring that the training data shares a similar

Table 14
Overall performance difference between representative and non-representative training datasets across encoding techniques and item pair type

Accuracy rate %				
	Item pair type			
Encoding technique	Non-enemy	Uncertain without key overlap	Uncertain with key overlap	Enemy
String-based vector-space model (VSM)	1.4	7.2	8.7	−2
Bi-encoder: without training	1.3	4.7	10.4	−4.5
Cross-encoder: without training	1.8	3.3	12.4	−5.8
Cross-encoder: with training	1.6	2.3	17	0.8
Bi-encoder: with training	1.3	3.6	24.6	1.9
False positive rate %				
	Item pair type			
Encoding technique	Non-enemy	Uncertain without key overlap	Uncertain with key overlap	Enemy
String-based vector-space model (VSM)	−1.4	−7.3	−20.5	−2.5
Bi-encoder: without training	−1.3	−4.7	−16.2	−1.8
Cross-encoder: without training	−1.8	−3.3	−19.2	−1.3
Cross-encoder: with training	−1.6	−2.3	−22.9	−2.8
Bi-encoder: with training	−1.3	−3.6	−26.4	−3.5
False negative rate %				
	Item pair type			
Encoding technique	Non-enemy	Uncertain without key overlap	Uncertain with key overlap	Enemy
String-based vector-space model (VSM)	0	0.2	11.7	4.5
Bi-encoder: without training	0	0	5.8	6.3
Cross-encoder: without training	0	0	6.8	7.1
Cross-encoder: with training	0	0	6	2
Bi-encoder: with training	0	0	1.8	1.6

structure with the test data, and increasing the training data size, can improve the accuracy of similarity index values for item pairs. Contrary to prior research [15], our study found that the cross-encoder did not outperform the bi-encoder; instead, the bi-encoder with training produced significantly better enemy identification results.

During the research, some labeling errors were discovered in item pairs within the enemy and uncertain groups, emphasizing the importance of SME review of the classification results. An iterative rotating strategy could be integrated into the training and testing process to identify and correct mislabeled item pairs in both datasets, potentially enhancing the performance of the encoding model and the ML classification outcome.

Overall, while this study does not advocate for complete automation of enemy identification when using sufficient and representative training data, the transformer-based approach clearly outperforms non-transformer methods in accurately representing the semantic meanings of item pairs, leading to better enemy identification results from supervised ML models. A follow-up study will incorporate more item pairs labeled by SMEs in the uncertain groups. Richer training data is anticipated to enable transformer models to better adapt to domain-specific content, producing more accurate similarity indices and thus improving the accuracy of the machine-based enemy identification approach.

Ethical Statement

This study does not contain any studies with human or animal subjects performed by any of the authors.

Conflicts of Interest

The authors declare that they have no conflicts of interest to this work.

Data Availability Statement

Data sharing is not applicable to this article as no new data were created or analyzed in this study.

Author Contribution Statement

Huijuan Meng: Conceptualization, Methodology, Software, Validation, Formal analysis, Investigation, Resources, Data curation, Writing – original draft, Writing – review & editing, Visualization, Supervision, Project administration. **Jinshu Li:** Validation, Investigation, Resources, Writing – review & editing, Project administration.

References

[1] Weir II, J. B. (2019). *Enemy item detection using data mining methods*. PhD Thesis, The University of North Carolina at Greensboro.

[2] Gibbons, R. D., Weiss, D. J., Frank, E., & Kupfer, D. (2016). Computerized adaptive diagnosis and testing of mental health disorders. *Annual Review of Clinical Psychology*, *12*, 83–104. https://doi.org/10.1146/annurev-clinpsy-021815-093634

[3] Ackerman, T. A., & Spray, J. A. (1986). *A general model for item dependency (ED272579)*. ERIC. https://files.eric.ed.gov/fulltext/ED272579.pdf

[4] Gomaa, W. H., & Fahmy, A. A. (2013). A survey of text similarity approaches. *International Journal of Computer Applications*, *68*, 13–18. https://doi.org/10.5120/11638-7118

[5] Wang, J., & Dong, Y. (2020). Measurement of text similarity: A survey. *Information*, *11*(9), 421. https://doi.org/10.3390/info11090421

[6] Kusner, M., Sun, Y., Kolkin, N., & Weinberger, K. (2015). From word embeddings to document distances. In *Proceedings of the 32nd International Conference on Machine Learning*, 957–966.

[7] Manning, C. D., & Schütz, H. (1999). *Foundations of statistical natural language processing*. USA: MIT Press.

[8] Deerwester, S., Dumais, S. T., Furnas, G. W., Landauer, T. K., & Harshman, R. A. (1990). Indexing by latent semantic analysis. *Journal of the American Society for Information Science*, *41*(6), 391–407. https://doi.org/10.1002/(SICI)1097-4571(199009)41:6<391::AID-ASI1>3.0.CO;2-9

[9] Blei, D. M., Ng, A. Y., & Jordan, M. I. (2003). Latent Dirichlet allocation. *Journal of Machine Learning Research*, *3*, 993–1022. https://dl.acm.org/doi/pdf/10.5555/944919.944937

[10] Pennington, J., Socher, R., & Manning, C. D. (2014). GloVe: Global vectors for word representation. In *Proceedings of the 2014 Conference on Empirical Methods in Natural Language Processing*, 1532–1543. https://doi.org/10.3115/v1/d14-1162

[11] Mikolov, T., Chen, K., Corrado, G. S., & Dean, J. (2013). Efficient estimation of word representations in vector space. *arXiv Preprint:1301.3781*.

[12] Vaswani, A., Shazeer, N., Parmar, N., Uszkoreit, J., Jones, L., Gomez, A. N., . . . , & Polosukhin, I. (2017). Attention is all you need. In *31st Conference on Neural Information Processing Systems.*

[13] Devlin, J., Chang, M., Lee, K., & Toutanova, K. (2018). BERT: Pre-training of deep bidirectional transformers for language understanding. *arXiv Preprint:1810.04805*.

[14] Radford, A., Narasimhan, K., Salimans, T., & Sutskever, I. (2018). *Improving language understanding by generative pre-training*. Retrieved from: https://cdn.openai.com/research-covers/language-unsupervised/language_understanding_paper.pdf

[15] Reimers, N., & Gurevych, I. (2019). Sentence-BERT: Sentence embeddings using Siamese BERT-Networks. In *Proceedings of the 2019 Conference on Empirical Methods in Natural Language Processing and the 9th International Joint Conference on Natural Language Processing*, 3982–3992. https://doi.org/10.18653/v1/d19-1410

[16] Fu, Y., & Han, K. (2022). Enemy item identification for different item types. In *Annual Meeting of the National Council on Measurement in Education*.

[17] Mao, X., Zhang, Q., & Clem, A. (2021). An exploration of an integrated approach for enemy item identification. *International Journal of Intelligent Technologies and Applied Statistics*, *14*(2), 123–134.

[18] Peng, F. (2020). *Automatic enemy item detection using natural language processing*. PhD Thesis, The University of Illinois at Chicago.

[19] Becker, K. A., & Kao, S. (2022). Identifying enemy item pairs using natural language processing. *Journal of Applied Testing Technology*, *23*, 41–52.

[20] Soyalp, G., Alar, A., Ozkanli, K., & Yildiz, B. (2021). Improving text classification with transformer. In *6th International Conference on Computer Science and Engineering*, 707–712. https://doi.org/10.1109/UBMK52708.2021.9558906

[21] Tezgider, M., Yildiz, B., & Aydin, G. (2022). Text classification using improved bidirectional transformer. *Concurrency and Computation: Practice and Experience*, *34*(9), e6486, https://doi.org/10.1002/cpe.6486

[22] Song, K., Tan, X., Qin, T., Lu, J., & Liu, T. (2020). *MPNet: Masked and permuted pre-training for language understanding*. *arXiv Preprint:2004.09297*.

[23] Tan, X. (2020). *MPNet combines strengths of masked and permuted language modeling for language understanding*. Retrieved from: https://www.microsoft.com/en-us/research/blog/mpnet-combines-strengths-of-masked-and-permuted-language-modeling-for-language-understanding/

[24] SBERT. (n.d.). *Pretrained models*. Retrieved from: https://www.sbert.net/docs/pretrained_models.html

[25] Kuhn, M., & Johnson, K. (2013). *Applied predictive modeling*. USA: Springer. https://doi.org/10.1007/978-1-4614-6849-3

How to Cite: Meng, H., & Li, J. (2024). Automatic Enemy Identification—Are We There Yet? *Artificial Intelligence and Applications*, *2*(4), 323–334. https://doi.org/10.47852/bonviewAIA42022424