Nuclear Threat Classifier

Andre Shportko
Northwestern University
andreshportko2026
@u.nortwestern.edu

Simon Sung
Northwestern University
simonsung2023
@u.nortwestern.edu

Roman Svintitskyy
Northwestern University
romansvintitskyy2023
@u.nortwestern.edu

Abstract

This paper investigates the development of a nuclear threat classifier using a few categorization techniques. The classifiers automatically categorize text data related to nuclear threats for efficient monitoring of nuclear security. It examines data collection, preprocessing, model configurations, assumptions, biases, and ethical considerations. The study contributes to accurate and responsible approaches to nuclear security.

1 Introduction

In the realm of the unprovoked 2022 Russian invasion of Ukraine, nuclear threats pose significant challenges to global security. By leveraging natural language processing and machine learning algorithms, we developed classifiers that can process large volumes of textual information and attempt to provide insights into nuclear security. The paper explores various aspects of the classifier, including data collection methodologies, preprocessing techniques, and model hyperparameters.

It also investigates the impact of different assumptions on the classifier's performance, shedding light on potential biases within the annotation process. Ethical considerations surrounding the use of such models in real-world scenarios are also discussed, emphasizing the importance of human judgment and decision-making in nuclear threat assessment. By evaluating the capabilities and limitations of the nuclear threat classifier, this research aims to contribute to the development of more accurate and reliable approaches to nuclear security, enhancing our ability to address and mitigate the risks associated with nuclear threats.

2 Data Collection and Preprocessing

2.1 Data Collection

We sourced our data from "Nuclear rhetoric and escalation management in Russia's war against Ukraine: A Chronology", a compendium of articles and videos about nuclear-related events from Russia and the West (Arndt and Horovitz, 2022). The chronology spans from the winter to the summer of 2022 and consists of 90 documents involving Russian activities (labeled ”RU") and Western reactions (labeled "W"). Each document in the chronology is also labeled as being either escalatory, warning, or de-escalatory before being briefly summarized in a condensed timeline at the beginning of the paper. The remaining portion fleshes out each event via a more comprehensive summary with citations to relevant news articles and videos.

We decided to train models on three different representations of each event, which we called *text-short*, *text-medium*, and *text*. For a given event, *text-short* represents the event's brief summary from the beginning of the paper, *text-medium* comes from the comprehensive summary in the main part of the paper, and *text* is the unabridged information from the primary source on which the other two texts are based.

We manually copied the content of the chronology into a spreadsheet, which we split according to the labels described above.

While recording *text-short* and *text-medium* were trivial, determining what to put into *text* was not as simple. Many of the events had multiple citations, in which case we arbitrarily picked a single source to transcribe into *text*. Occasionally, the only cited source for an event was a video. In these cases, we copied the text in *text-medium* over to *text*, which we also did when we ran into a paywall. For articles that were written in Russian or French, we used Google Translate for the sake of time.

We soon realized that, of the 90 events, 11 were labeled “escalatory”, 37 as ”de-escalatory”, and 42 as a ”warning”, which makes the dataset imbalanced.

2.2 Data Preprocessing

We conducted some preprocessing on our data in order to correct some issues with our data. We first did some exploratory data analysis (EDA), checked for NaNs and other nonsensical values, and converted our data to appropriate data types in order to reaffirm the quality of our data.

We then generated a new dataset using the pandas and NumPy to split each data point into sentences and created a temporary dataframe that consisted only of the sentences in a given data entry, each of which was labeled as ”E”, ”D”, or ”W” (escalatory, de-escalatory, warning) for the label and either ’r’ or ’w’ (Russia, West).

Splitting each data point into sentences resulted in turning 90 events into over 6,800 data entries. Each sentence’s label inherits the label of the original event. One thing to note about this approach is that this may have resulted in a bit of noise; certain sentences that may be considered relatively neutral or meaningless might have been labeled as ’E’, ’W’, or ’D’.

3 Support Vector Machine (SVM) Model

3.1 Description of the Model

Support vector machine models were a kind of model we wanted to explore for the purpose of this assignment. This particular model piqued our interest due to its reputation for being a powerful classification algorithm that performs well with binary classification but may struggle with multi-class classification tasks.

3.2 Methods

Before training the model, we conducted EDA by creating bar plots with ’side’ and ’label’ counts. This made it clear that, as described in Section 2, the data was imbalanced. We addressed this issue by upsampling the minority class and/or downsampling the majority class (Li, 2018), (Mazumder, 2021). In the end, we decided that it would be interesting to measure the performance of our model using both the unbalanced data and the adjusted version on both features of interest (’side’ and ’label’).

Since SVM models cannot accept the text as input, we had to refactor our dataset of sentences into a format that would fit the SVM model - numerical vectors. To do so, we used a TF-IDF vector representation. Then, we followed the standard data pipeline:

- Defined the features to train on and the target variable
- Did an 80/20 train/test split with sklearn.train_test_split()
- Initialized the sklearn.LinearSVC() as our SVM
- Trained (fit) the model on our train data
- Made predictions with our test data
- Trained (fit) the model on our train data
- Evaluated their performance by considering precision, recall, f-1 score, accuracy, and weighted average.

3.3 Results and Analysis

All three datasets were imbalanced. The largest data set consisted of approximately 1300 ”E” entries, 2400 ”W” entries, and 3200 ”D” entries. Similar proportions were observed in the other data sets as well. The ”side” label was more evenly split across the board. Balancing the data sets with respect to ”label” didn’t do much to change the ratios. One interesting thing we observed was that, after upsampling and downsampling the dataset, the ratio stayed roughly the same but it went from containing slightly more ”w” entries to having slightly more ”r” entries.

As we expected, our SVM models performed better on balanced datasets compared to imbalanced ones. Since SVMs typically perform better with binary classification tasks compared to multi-class classification, we expected our model to perform better when it was trained on ”side” rather than

”label”. This turned out to be the case. The experimental results for the largest dataset can be viewed below.

Table 1: ”label”: imbalanced dataset

	precision	recall	f1	support
W	0.63	0.68	0.65	636
E	0.49	0.38	0.43	263
D	0.55	0.56	0.56	463
acc.			0.58	1362
m. avg	0.56	0.54	0.55	1362
w. avg	0.58	0.58	0.58	1362

Table 2: ”label”: balanced dataset

	precision	recall	f1	support
W	0.68	0.71	0.69	467
E	0.76	0.81	0.78	497
D	0.70	0.61	0.65	467
acc.			0.71	1431
m. avg	0.71	0.71	0.71	1431
w. avg	0.71	0.71	0.71	1431

Table 3: ”side”: imbalanced dataset

	precision	recall	f1	support
r	0.79	0.78	0.78	665
w	0.79	0.80	0.80	697
acc.			0.79	1362
m. avg	0.79	0.79	0.79	1362
w. avg	0.79	0.79	0.79	1362

Table 4: ”side”: balanced dataset

	precision	recall	f1	support
r	0.87	0.89	0.88	746
w	0.88	0.86	0.87	685
acc.			0.87	1431
m. avg	0.88	0.87	0.87	1431
w. avg	0.87	0.87	0.87	1431

4 Results of the Spacy TextCategorizer model

4.1 Description of the Model

The nuclear threat classifier model utilizes a convolutional neural network (CNN) for text categorization (Leonard, 2020), trained using the Spacy framework. (Boyd, 2020)

For this task, we preprocessed the sentences by removing stop words (except for ”no” and ”not”), removing punctuation, and lemmatizing words.

For evaluation purposes, the true event is considered when the model's prediction aligns with the test label, and the positive event is defined as ”Escalation”. The following definitions apply for the evaluation metrics: TP (true positive) indicates that the model correctly predicted an escalatory event, FP (false positive) indicates that the model incorrectly predicted an escalation, TN (true negative) denotes the model's correct prediction of a non-escalatory event (Warning or De-escalation), and FN (false negative) represents the model's incorrect prediction of a non-escalatory event.

To check if noise from the sentence-level approach is too distracting, we added a hyperparameter ”expand”. If expand=True, the model will be trained on the data split by sentences.

Other model's hyperparameters are set with n_epochs = 8 (the number of training epochs.) train_test_split is 90-to-10

4.2 Binary Assumption

The model is designed to support trinary classification, allowing text to be categorized into three distinct labels or categories. However, there is an additional parameter called ”assume” that can be set to a tuple (X_1, X_2) to switch the model to binary classification. Here, X_1 represents the old label, and X_2 represents the new label.

This adjustment is achieved by making an assumption that simplifies the original three-label classification task into a binary one. The term ”assumption” refers to the reduction of complexity in the classification task.

By training the model on this data and activating the "assume" parameter by a tuple (W, E), the "W" value will be treated as "E". This assumption simplifies the task by collapsing the trinary classification into binary classification.

4.3 No Assumption: West

Refer to the tables 5, 6, 7.

Table 5: No Assumption: West (short text)

Expand	F1	Accuracy	TP/TN/FP/FN
True	0.0	0.67	(0, 4, 1, 1)
False	1.0	1.0	(1, 4, 0, 0)

Table 6: No Assumption: West (medium text)

Expand	F1	Accuracy	TP/TN/FP/FN
True	0.0	0.89	(0, 25, 1, 2)
False	0.0	0.8	(0, 4, 0, 1)

Based on the results, it is evident that the expand=True option should always be chosen due to an extreme deficit of data, and the train-test split is not adequate for valid calculations.

Table 7: No Assumption: West (full text)

F1	Accuracy	TP/TN/FP/FN
0.40	0.86	(16, 284, 14, 35)

Additionally, for the lack of data, the full "text" input should always be chosen. Although the "more data = better" conclusion sounds trivial, it is not immediately obvious. A lot of sentences from "escalatory" articles are peaceful (f.e., "not work otherwise") which leads to more FNs on the testing.

4.4 No Assumption: Russia

Refer to table 8.

Table 8: No Assumption: Russia

F1	Accuracy	TP/TN/FP/FN
0.45	0.77	(30, 217, 41, 31)

4.5 No Assumption: Side-Blind

Refer to table 9.
It is evident that the side-blind assumption should not be chosen, and each model should be trained specifically for each side. Although there is a small increase in accuracy, this is primarily due to the high number of true negatives and imbalanced data.

Table 9: No Assumption: Side-Blind

F1	Accuracy	TP/TN/FP/FN
0.37	0.81	(38, 513, 56, 72)

4.6 (W, E) assumption

Refer to table 10.

Table 10: (W, E) assumption

Side	F1	Accuracy	TP/TN/FP/FN
West	0.58	0.67	(80, 155, 57, 57)
Russia	0.84	0.75	(208, 32, 56, 23)

The drastic 26% increase in F1 score when the "side" was switched implies that assuming Russia's "warnings" as "escalations" leads to more consistent and reproducible results compared to assuming West's "warnings" as "escalations." This might imply a bias in the annotation.

4.7 (W, D) assumption

Refer to table 11.

Table 11: (W, D) assumption

Side	F1	Accuracy	TP/TN/FP/FN
West	0.33	0.81	(16, 267, 31, 35)
Russia	0.38	0.76	(23, 220, 38, 38)

The drastic drop in the F1 score compared to the (W, E) assumption (Table 10) suggests that assuming a "warning" means "de-escalation" is misleading for the model, regardless of the side.

4.8 Error Analysis

We discovered that the Spacy text classification model may have run into issues with handling negation (Mehta, 2022). This may have caused phrases that were deescalatory in context to be marked as escalatory, leading to false positives.

We notice that it was common for false positives to contain "buzzwords" that don't really imply anything without context. For example, sentences that contain "deterrence" and "risk" several times, but are possibly marked as escalatory due to the influence of words like "military" and "policy".

Another thing pattern we noticed is that various words may ”sound” escalatory, but are not particularly relevant to nuclear activities. For example, contains words like ”horror”, ”damage”, and ”soldier”, which seem to be more related to the ongoing war than to nuclear escalation/de-escalation itself.

4.9 Quintagrams associated with Russian Escalation

We generated the set of Quintagrams from the entire dataset. Based on the (W, E) assumption on the Russian side, consider the following 7 Quintagrams within the top 50 that are associated with Nuclear Threat Escalation (Table 12). The syntax and agreement are distorted due to the removal of stop words and lemmatization.

To reproduce results:
$n\_epochs = 10$
$assume = (W, E)$
$side = r$

Table 12: Russian Quintagrams

Quintagram	Probab.
everyone russia everyone blaming russia	1.0000
totalitarian regime outwardly looked wonderful	1.0000
west would provided endless support	1.0000
deputy come go deep aggressive	0.9999
whole suffered major blow trust	0.9999
done blatantly stated united state	0.9999
lost industrial technological potential –	0.9999

5 Ethical Considerations and Conclusion

The usage of any of our models for real-life purposes is ethically questionable. The paper concludes that the model was trained on biased data. Moreover, if the model’s purpose is to enable offensive actions or facilitate preemptive strikes, it raises the risk of conflict escalation and human suffering. While a nuclear threat estimator model could provide valuable insights, it should not replace human decision-making. Ultimately, decisions regarding nuclear threats should involve careful deliberation by human experts, considering various factors such as political, diplomatic, and humanitarian considerations.

Assessing the potential risks and unintended consequences, such as accidental escalation or false alarms, is vital to mitigate harm and ensure the model’s ethical use.

6 Task Distribution and GitHub Repository

Our team held regular meetings to decide on our project idea and to confirm that things were going smoothly. Each member of our team manually transcribed the corresponding instances of *text-short*, *text-medium*, and *text* for 30 events into a spreadsheet for data collection purposes. We all worked on a unique model. Andre worked on the Spacy TextCategorizor model, Roman worked on the SVM models, and Simon attempted to train a model based on TensorFlow’s Keras API (we weren’t very happy with the quality of the results, so we decided to omit this one from this report). For each model, we handled the data preprocessing and contributed to the paper.

The full dataset and Google Colab notebooks for each model are available on GitHub (Shportko et al., 2023).

References

Anna Clara Arndt and Dr Liviu Horovitz. 2022. Nuclear rhetoric and escalation management in russia’s war against ukraine: A chronology. *RESEARCH DIVISION INTERNATIONAL SECURITY*.

Adriane Boyd. 2020. Spacy textcategorizer.

Mat Leonard. 2020. Text classification.

Susan Li. 2018. Multi-class text classification with scikit-learn.

Saikat Mazumder. 2021. 5 techniques to handle imbalanced data for a classification problem.

Sourabh Mehta. 2022. When to use negation handling in sentiment analysis?

Andre Shportko, Simon Sung, and Roman Svintitskyy. 2023. Github repository of the nuclear threat classifier.