SuteAlbastre at SemEval-2024 Task 4: Predicting Propaganda Techniques in Multilingual Memes using Joint Text and Vision Transformers

Ion-Marian Anghelina, Gabriel-Sebastian Buță and **Alexandru Enache**
University of Bucharest
Faculty of Mathematics and Computer Science
{ion.anghelina, gabriel.buta, alexandru.enache1}@s.unibuc.ro

Abstract

The main goal of this year's SemEval Task 4 is detecting the presence of persuasion techniques in various meme formats. While **Subtask 1** targets text-only posts, **Subtask 2**, subsections **a** and **b** tackle posts containing both images and captions. The first 2 subtasks consist of **multi-class** and **multi-label** classifications, in the context of a **hierarchical** taxonomy of 22 different persuasion techniques.

This paper proposes a solution for persuasion detection in both these scenarios and for various languages of the caption text. Our team's main approach consists of a Multimodal Learning Neural Network architecture, having Textual and Vision Transformers as its backbone. The models that we have experimented with include EfficientNet and ViT as visual encoders and BERT and GPT2 as textual encoders.

1 Introduction

In nowadays society, the role of social media in opinion formation is more important than ever. A fundamentally form of social media leisure, the meme has become a powerful resource which can easily be abused by various entities with political interests. The most well known platforms have a strict policy regarding misleading information, especially of political nature. However, posts containing such information are hard to automatically detect, and the administrators mostly rely on user reports.

This paper proposes an automatic detection solution for Arabic, Bulgarian, English and North Macedonian, suitable for both text-only and text-image memes. (Dimitrov et al., 2024)

Only English training data was provided for all the Subtasks, all the other languages' tasks requiring a One-Shot Learning approach.

The proposed model excels on **Subtask 2a**, scoring 2^{nd} place for all languages, besides English and also on **Subtask 2b** where it also ranked second, achieving an $F1$-score of over 0.8 in the binary setup. It struggled, however, on **Subtask 1**, especially in the case of the Subtask variants for languages without training samples, achieving an $F1$-score of about 0.2 on average.

The full results table can be found in the **Results** subsection in Table 6.

Python code for all the used models and algorithms is available in our GitHub Repository.

2 Background

2.1 Related Work

Dimitrov et al. in "Detecting propaganda techniques in memes" have also conducted their own approach of solving the **Subtask 1** and **Subtask 2b** on their own dataset in a Multimodal setup (Dimitrov et al., 2021).

Martino et al. in "A Survey on Computational Propaganda Detection" reviewed the state of the on computational propaganda detection from both the perspective of using Natural Language Processing in order to detect propaganda, as well as analysing users profiles in order to detect a propaganda networks on media platforms (Martino et al., 2020). They tackle **Subtask 1** and **Subtask 2b** as well, being, however, asked to also provide all the occurences' positions for each propaganda technique.

2.2 Input

The text information is given as a JSON which, for each meme, it has a unique **id** assigned, a **text** representing the text that is written on the meme. For the **Subtask 2a** and **Subtask 2b** we also have a **image** attribute representing the name of the image to which the previously mentioned information is corresponding.

Proceedings of the 18th International Workshop on Semantic Evaluation (SemEval-2024), pages 443–449

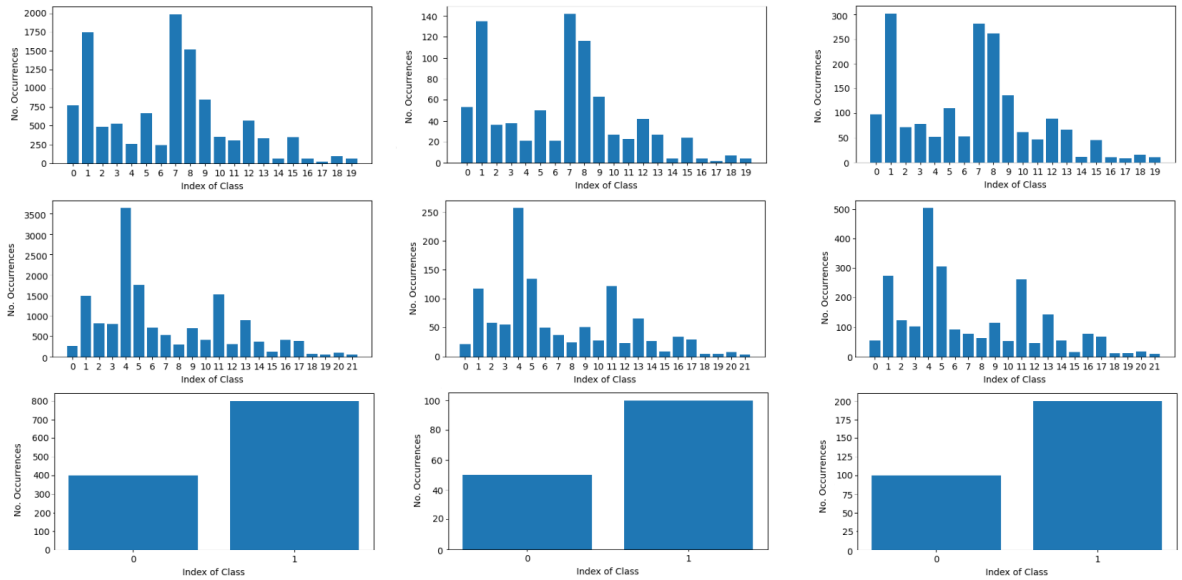

Figure 1: The class distribution for **Subtask 1**, **Subtask 2a** and respectively **Subtask 2b** (on rows) for the **train**, **validation** and respectively **dev** sets (on columns)

Below is an example extracted from the English test dataset for **Subtask 1**:

```
{
    "id": "64747",
    "text": "Just a few of the Rino's
        that need to go!!\\nRino season
        will be open soon!"
}
```

And another example from the English test dataset for **Subtask 2a**:

```
{
    "id": "79142",
    "text": "NOW ENTERING 2022",
    "image": "prop_meme_24023.png"
}
```

2.3 Output

For the **Subtask 1** and **Subtask 2a**, the output is returned in a JSON file which, for each meme, contains the unique **id** through which the photo was identified in the input, as well as a **labels** list containing the

labels corresponding to the meme. For **Subtask 2b**, instead of the labels list we will have only a **label** that is represented either by the string **propagandistic** or **non_propagandistic**. Below is an example extracted from a submission for the **Subtask 1** of the Arabic test dataset:

```
{
    "id": "00001",
    "labels": ["Black-and-white Fallacy/
        Dictatorship", "Presenting
        Irrelevant Data (Red Herring)"]
}
```

And another example from the Arabic test dataset for **Subtask 2b**:

```
{
    "id": "00007",
    "label": "non_propagandistic"
}
```

2.4 Dataset

The dataset is composed of memes with English captions present on them. For **Subtask 1** and **Subtask 2a**, we have **7000** train images, **500** validation images and **1000** dev images, while for **Subtask 2b**, we have **1200** train images, **150** validation images and **300** dev images.

From **Figure 1** we can observe that the class distribution is conserved throughout the train, validation and dev sets.

2.5 Datasets used

For the **English** task, for **Subtask 1**, **Subtask 2a** and **Subtask 2b**, we fine-tuned **limjiayi/bert-hateful-memes-expanded** (limjiayi, 2024) which is a model based on **bert-base-uncased** (Devlin et al., 2018a) which was previously fine-tuned on **HatefulMemes** (Kiela et al., 2020), **HarMemes** (Dimitrov, 2024) and **MultiOff**(Suryawanshi et al., 2020) datasets.

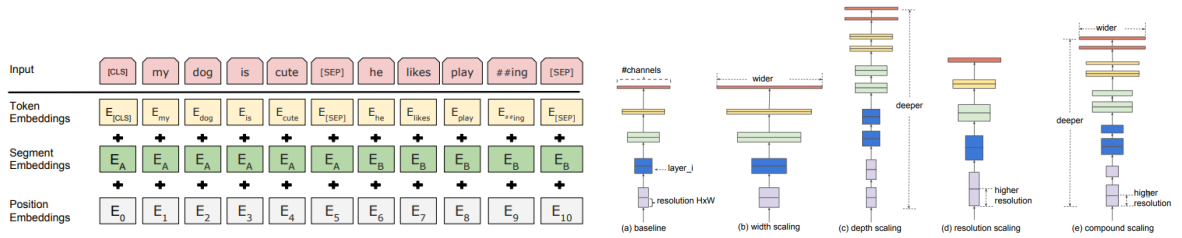

Figure 2: On the left side we have BERT Text Encoder Visual Representation (Devlin et al., 2018a) while on the right side we have the Model Scaling of an EfficientNet model (Tan and Le, 2019)

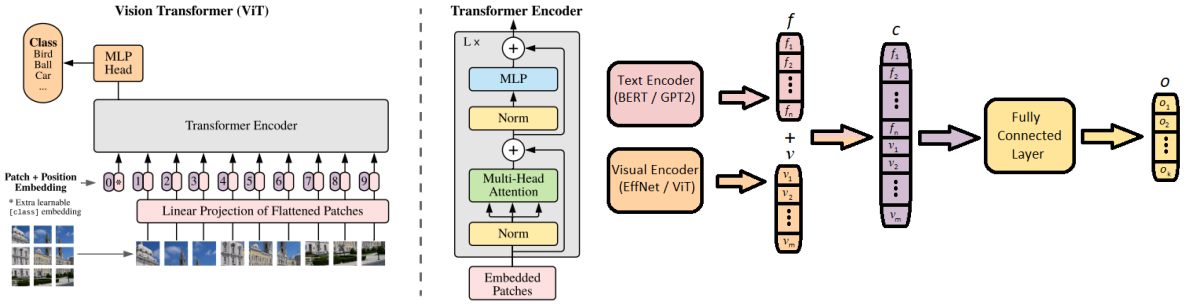

Figure 3: On the left side we have the model overview of a Vision Transformer (Radford et al., 2019) while on the right side we have our architecture of solving the proposed task

For the **Bulgarian** task, for **Subtask 1**, **Subtask 2a** and **Subtask 2b**, we used **usmiva/bert-web-bg** model (Marinova et al., 2023), an architecture pretrained from scratch BERT on Bulgarian dataset created at the Bulgarian Academy of Sciences.

For the **North Macedonian** task, for **Subtask 1**, **Subtask 2a** and **Subtask 2b**, we used **macedonizer/mk-gpt2** (Radford et al., 2019) model which is a model based on GPT2-large which was trained on English language using a causal language modeling.

For the **Arabic** task, for **Subtask 1** and **Subtask 2b**, we used **asafaya/bert-base-arabic** (Safaya et al., 2020) model which is a model based on BERT base fine-tuned on the arabic corpus. For **Subtask 2a** we used the **google-bert/bert-base-multilingual-uncased** model (Devlin et al., 2018b).

3 System Overview

3.1 Architecture Overview

For **Subtask 1**, our approach consists of using a text encoder which provides an accurate vectorial embedding of the memes captions. These representations are later used to train a **Fully Connected Neural Network**. The latter network's last layer consists of 20 **output neurons**, each representing the probability of existence of one of the 20 persuasion techniques.

For **Subtasks 2a and 2b**, the input format requires a special model branch for preprocessing the images. Similar to the **text encoder**, the **image encoder** outputs a dense feature vector representing a spacial embedding of the image. The results of the two encoders are used to train a similar Fully Connected Neural Network. In the case of **Subtask 2a**, the last layer of the neural network consist of 22 **output neurons**, each representing the probability of finding a certain persuasion technique. For subtask 2*b*, however, the last layer represents only 2 neurons, whose outputs are the probabilities of the text containing a persuasion technique or not.

While **Subtasks 1 and 2a** are treated as regression tasks with 20 and 22 output values, respectively, **Subtask 2b** is treated as a simple binary classification task.

3.2 Textual Encoders

For the text encoder branch of the model, pretrained variants of the following transformers were used:

- **BERT**(Devlin et al., 2018a) (Figure 2) is a very powerful text representation model, since it can capture large bidirectional links between words in order to build accurate word embeddings. It

relies on the self attention mechanism.

- **RoBERTa** (Liu et al., 2019) is an improved version of **BERT**, using a dynamic masking strategy during training, which is conducted over larger datasets and with a larger batch-size.

- **GPT-2** (Radford et al., 2019) is a transformer based model, pretrained on various dataset, which excels in the task of text generation based on a given prompt, but which can also be used as a backbone in any classification or regression task.

3.3 Visual Encoders

- **Vision Transformers** (Dosovitskiy et al., 2020) (Figure 3) are a class of Computer Vision models which, unlike their predecessors, do not rely on Convolutional Neural Networks as their backbone, but utilize the **Transformer** architecture, adapted from **NLP** tasks. It bears a resemblance to **BERT**, the main difference being that patches of the image are used instead of words. It was pretrained on the **ImageNet_21k** dataset. (Ridnik et al., 2021)

- **EfficientNet** (Tan and Le, 2019) (Figure 2) is a class of Convolutional Neural Network models which achieve high performance on image classification tasks. They mainly rely on automatically scaling the depth and width the network with respect to fixed parameters regarding the dataset to obtain the best possible results.

3.4 Predicting the answer

For **Subtask 1**, we denote $h_{i,j}$, the output of the **last hidden layer** of the textual encoder model the j^{th} token of the i^{th} sample.

Every sample in the training set has a fixed dimension of t tokens, and each token is embedded into a 768 dimensional vector. The fully connected neural network has a hidden layer of size d.

We first flatten $h_{i,j}$, obtaining f, a dense vector of embeddings for each word.

If we denote by (W, b) the tensor of weights between the transformer output layer and the output layer of the network, and the biases of the output layer respectively, the output of the output layer will be represented by:

$$W \cdot \tanh(f) + b$$

The output layer has 20 neurons. The result for each of them is computed by applying the $Sigmoid$ activation function, which outputs a probability.

Thus, the final output of the model is:

$$Sigmoid(W \cdot \tanh(f) + b)$$

For **Subtasks 2a and 2b**, the dense feature vector f is defined similarly, representing the textual extracted features.

We define similarly v, the vector of features extracted by the **Visual Encoder** and c, the vector obtained by concatenating f and v.

$$c = fv$$

For **Subtask 2a**, the output of the model is very similar to the one of **Subtask 1**, the main difference being that the output layer now has 22 neurons. (Figure 3)

The output of this model is:

$$Sigmoid(W \cdot tanh(c) + b)$$

For **Subtask 2b**, the output of the model is a probability distribution, the 2 classes of the binary classification being dependent of each other. We can obtain such an output using the $Softmax$ activation function.

The output is, thus:

$$Softmax(W \cdot tanh(c) + b)$$

Where W and b are defined similarly to **Subtask 1**.

The outputs of the model for **Subtasks 2a** and **2b** are, then, compared to a threshold, which represents the lower limit of the probability for a persuasion technique to be considered used.

3.5 Transfer Learning for the Visual Encoder

During the competition, the teams were not provided training datasets in **Bulgarian, North Macedonian or Arabic**. An interesting approach we tried was using the **English** labeled data for training our model and, after that, using the resulting **Vision Encoder** component of this model in conjunction with an adequate pretrained **Textual Encoder** for the desired language.

Using this approach, we made use of the image information in the dataset, which is agnostic to the language of the meme.

3.6 Adaptive Thresholding

Since we are treating **Subtask 1** and **Subtask 2a** as regression tasks (our models output a number between 0 and 1 for each persuasion technique), we needed to choose a threshold which determines the minimum value such that a persuasion technique should be included in the answer or not. In our final submissions, we have chosen **0.25** as the threshold for all the techniques. We have determined by experimenting on the dev dataset that this threshold maximizes the Hierarchical F1 score by creating a balance between the Hierarchical Precision and the Hierarchical Recall.

One idea which we tested only after the competition was to have a separate threshold for each persuasion technique. The method used to determine this was by using the validation dataset in order to find the threshold which maximizes the F1 score for that technique. An efficient way to find the best threshold is to sort the predictions and start with a threshold of 0. This would mean that all the samples are considered true, so $TP = sum(GT)$, $FP = N - sum(GT)$ and $FN = 0$, where N is the number of samples and GT is the array of ground truths. We can calculate the $F1$ score using the formula $2TP/(2TP+FP+FN)$. Now we go though all the prediction in order and assume that the current threshold is the value of the prediction. This means that the current sample i is now considered false, so if $GT_i = 1$, then $TP = TP - 1$ and $FN = FN + 1$, else $FP = FP - 1$. This allows to compute the current $F1$ score in $O(1)$.

Also, we have found that sometimes this algorithm would find very low values, so we have limited the thresholds to 0.2 as the minimum value.

Table 1: Results after the competition using adaptive thresholding on the test dataset

Subtask	F1-Score
1 English	0.647
2a **English**	**0.695**

This method performed worse for **Subtask 1** compared to our final submission (0.657), but performed better for **Subtask 2a**, where it improved our final result of 0.684 to 0.695. (Table 1)

We also tested a smaller threshold of 0.2 for the **Arabic Subtask 2a** which managed to get 1^{st} place on the final standings with a score of 0.585.

4 Experimental Setup

For the training of models we have used the data splits in the following manner. Before the dev gold labels were available, we have used the validation dataset in order to find the best hyperparameters and after that added the validation dataset to the training data. After the dev gold labels were published, in order to submit on the test dataset, we have also trained on the dev dataset. This maximized the number of training samples available.

The preprocessing techniques used for the images are:

- Resizing the image to 224×224 pixel size to match pretraining size for the Vision Transformer
- Scaling the values on all color chanels with a standard distribution $\mathcal{N}(0.5, 0.5^2)$ to increase numerical stability and facilitate learning

The preprocessing techniques used for the texts are:

- Lowercasing all characters, necessary especially in the context of memes which do not follow a casing norm
- Tokenizing the texts into representative tokens using the pretrained tokenizer associated with the used model
- Padding or truncating the texts to a standard dimension of 128 for the transformer model

All the external libraries (and their versions) used for the setup are listed in our GitHub Repository in the requirements.txt file.

The experiments conducted can be seen in tables 2, 3 and 4.

Epochs ALL	Epochs FC	Thresh	hP	hR	hF_1
5	5	0.25	0.62	0.60	0.612
3	3	0.25	0.63	0.66	**0.646**

Table 2: Experiments for **Subtask 1** on the dev dataset

Epochs ALL	Epochs FC	Thresh	hP	hR	hF_1
3	3	0.5	0.77	0.52	0.62
3	3	0.25	0.69	0.66	0.680
3	0	0.3	0.70	0.66	0.684
3	0	0.25	0.67	0.70	**0.687**

Table 3: Experiments for **Subtask 2a** on the dev dataset

Epochs ALL	Epochs FC	F_1 $macro$	F_1 $micro$
3	0	0.735	0.760
3	3	0.754	0.773
5	5	**0.819**	0.846

Table 4: Experiments for **Subtask 2b** on the dev dataset

Subtask	Visual Encoder	Text Encoder	Epochs ALL	Epochs FC
1 English	ViT	BERT	3	3
2a English	ViT	BERT	3	3
2b English	ViT	BERT	5	5
1 Bulgarian	ViT	BERT	3	3
2a Bulgarian	ViT	BERT	3	0
2b Bulgarian	ViT	BERT	0	5
1 N. Macedonian	ViT	GPT2	3	3
2a N. Macedonian	ViT	GPT2	3	0
2b N. Macedonian	ViT	GPT2	5	5
1 Arabic	ViT	BERT	3	3
2a Arabic	ViT	BERT	0	3
2b Arabic	ViT	BERT	0	5

Table 5: The configurations of our final submissions on the test dataset

In Table 5, by "Epochs ALL" and "Epochs FC" we refer to the epochs for which the whole model is trained, respectively to the epochs where only the last fully connected layers are trained and the encoders are frozen. In the case of the non-english subtasks, the text encoder was always frozen during the training. In the cases where "Epochs ALL" appears as 0, we have experimented with taking the trained encoder from the corresponding english subtask and freezing it in the training process.

The first task we approached was **Subtask 2b**. We first used varieties of the **EfficientNet** model for the Visual Encoder, later switching to **ViT** encoding thanks to its higher performance.

The metrics used for **Subtask 1** and **Subtask 2a** are Hierarchical Precission (hP), Hierarchical Recall (hR) and Hierarchical F1 score (hF_1). In the multi-label settings, we define $\hat{C}_i$ as the set of ground truth classes and all their ancestors and $\hat{C}'_i$ as the set of predicted classes and all their ancestors (Kiritchenko et al., 2006). Thus the metrics are represented by :

$$hP = \frac{\sum_i |\hat{C}_i \cap \hat{C}'_i|}{\sum_i |\hat{C}'_i|} \quad hR = \frac{\sum_i |\hat{C}_i \cap \hat{C}'_i|}{\sum_i |\hat{C}_i|}$$

$$hF_1 = \frac{2 \cdot hP \cdot hR}{hP + hR}$$

For **Subtask 2b**, the metrics used are the classic F_1 $macro$ and F_1 $micro$.

5 Results

Subtask	F1-Score	Place
1 English	0.657	9th
2a **English**	**0.684**	**5th**
2b **English**	**0.809**	**2nd**
1 Bulgarian	0.235	19th
2a **Bulgarian**	**0.610**	**2nd**
2b Bulgarian	0.594	7th
1 N. Macedonian	0.203	19th
2a **N. Macedonian**	**0.575**	**2nd**
2b N. Macedonian	0.177	14th
1 Arabic	0.234	16th
2a **Arabic**	**0.516**	**2nd**
2b Arabic	0.500	10th

Table 6: Results during the competition on the test dataset

As it can be seen from the results tables 5 and 6, the best approach, in terms of fully training our model, rather than only fine-tuning the final classification layer, was varied. The latter approach being more suitable, especially for **Subtasks 2a** and **2b** in the case of languages without additional training data.

Our model managed to outperform the Competition Baseline for **Subtasks 2a** and **2b** for all the provided languages.

In the case of **Subtask 1**, our model only outperformed the Competition Baseline for the English language dataset.

6 Conclusion

Our model accurately creates dense embeddings for both the memes and their captions, managing to make use of their most prominent features in computing the result. The performance peaks on **Subtask 2a**. The Transformer components used in the proposed architectures are capable of learning from vast datasets with low risk of overfitting. Thus, one way of improving the solution would be the use of additional datasets related to the subject, in various languages. Last but not least, our future work will revolve around decision making on the **Hierarchical DAG**, further making use of relationship between different labels.

7 Acknowledgments

We would like to especially thank our colleague and collaborator Adrian-Ștefan Miclăuș (University of Bucharest), for continuously reviewing our work and offering valuable advice.

References

Jacob Devlin, Ming-Wei Chang, Kenton Lee, and Kristina Toutanova. 2018a. BERT: pre-training of deep bidirectional transformers for language understanding. *CoRR*, abs/1810.04805.

Jacob Devlin, Ming-Wei Chang, Kenton Lee, and Kristina Toutanova. 2018b. BERT: pre-training of deep bidirectional transformers for language understanding. *CoRR*, abs/1810.04805.

Dimitar Dimitrov. 2024. bert-hateful-memes-expanded. https://github.com/di-dimitrov/harmeme.

Dimitar Dimitrov, Firoj Alam, Maram Hasanain, Abul Hasnat, Fabrizio Silvestri, Preslav Nakov, and Giovanni Da San Martino. 2024. Semeval-2024 task 4: Multilingual detection of persuasion techniques in memes. In *Proceedings of the 18th International Workshop on Semantic Evaluation*, SemEval 2024, Mexico City, Mexico.

Dimitar Dimitrov, Bishr Bin Ali, Shaden Shaar, Firoj Alam, Fabrizio Silvestri, Hamed Firooz, Preslav Nakov, and Giovanni Da San Martino. 2021. Detecting propaganda techniques in memes. *arXiv preprint arXiv:2109.08013*.

Alexey Dosovitskiy, Lucas Beyer, Alexander Kolesnikov, Dirk Weissenborn, Xiaohua Zhai, Thomas Unterthiner, Mostafa Dehghani, Matthias Minderer, Georg Heigold, Sylvain Gelly, et al. 2020. An image is worth 16x16 words: Transformers for image recognition at scale. *arXiv preprint arXiv:2010.11929*.

Douwe Kiela, Hamed Firooz, Aravind Mohan, Vedanuj Goswami, Amanpreet Singh, Pratik Ringshia, and Davide Testuggine. 2020. The hateful memes challenge: Detecting hate speech in multimodal memes. *CoRR*, abs/2005.04790.

Svetlana Kiritchenko, Richard Nock, and Fazel Famili. 2006. Learning and evaluation in the presence of class hierarchies: Application to text categorization. volume 4013, pages 395–406.

limjiayi. 2024. bert-hateful-memes-expanded. https://huggingface.co/limjiayi/bert-hateful-memes-expanded.

Yinhan Liu, Myle Ott, Naman Goyal, Jingfei Du, Mandar Joshi, Danqi Chen, Omer Levy, Mike Lewis, Luke Zettlemoyer, and Veselin Stoyanov. 2019. Roberta: A robustly optimized bert pretraining approach. *arXiv preprint arXiv:1907.11692*.

Iva Marinova, Kiril Simov, and Petya Osenova. 2023. Transformer-based language models for bulgarian. In *Proceedings of the 14th International Conference on Recent Advances in Natural Language Processing*, pages 712–720.

Giovanni Da San Martino, Stefano Cresci, Alberto Barrón-Cedeño, Seunghak Yu, Roberto Di Pietro, and Preslav Nakov. 2020. A survey on computational propaganda detection. *arXiv preprint arXiv:2007.08024*.

Alec Radford, Jeffrey Wu, Rewon Child, David Luan, Dario Amodei, Ilya Sutskever, et al. 2019. Language models are unsupervised multitask learners. *OpenAI blog*, 1(8):9.

Tal Ridnik, Emanuel Ben-Baruch, Asaf Noy, and Lihi Zelnik-Manor. 2021. Imagenet-21k pretraining for the masses. *arXiv preprint arXiv:2104.10972*.

Ali Safaya, Moutasem Abdullatif, and Deniz Yuret. 2020. KUISAIL at SemEval-2020 task 12: BERT-CNN for offensive speech identification in social media. In *Proceedings of the Fourteenth Workshop on Semantic Evaluation*, pages 2054–2059, Barcelona (online). International Committee for Computational Linguistics.

Shardul Suryawanshi, Bharathi Raja Chakravarthi, Mihael Arcan, and Paul Buitelaar. 2020. Multimodal meme dataset (multioff) for identifying offensive content in image and text. In *Proceedings of the Second Workshop on Trolling, Aggression and Cyberbullying (TRAC-2020)*. Association for Computational Linguistics.

Mingxing Tan and Quoc Le. 2019. Efficientnet: Rethinking model scaling for convolutional neural networks. In *International conference on machine learning*, pages 6105–6114. PMLR.