Detecting Sockpuppetry on Wikipedia Using Meta-Learning

Luc Raszewski
The University of Melbourne
lraszewski@student.unimelb.edu.au

Christine De Kock
The University of Melbourne
christine.dekock@unimelb.edu.au

Abstract

Malicious sockpuppet detection on Wikipedia is critical to preserving access to reliable information on the internet and preventing the spread of disinformation. Prior machine learning approaches rely on stylistic and meta-data features, but do not prioritise adaptability to author-specific behaviours. As a result, they struggle to effectively model the behaviour of specific sockpuppet-groups, especially when text data is limited. To address this, we propose the application of meta-learning, a machine learning technique designed to improve performance in data-scarce settings by training models across multiple tasks. Meta-learning optimises a model for rapid adaptation to the writing style of a new sockpuppet-group. Our results show that meta-learning significantly enhances the precision of predictions compared to pre-trained models, marking an advancement in combating sockpuppetry on open editing platforms. We release a new dataset of sockpuppet investigations to foster future research in both sockpuppetry and meta-learning fields.

1 Introduction

Over recent years, social media sites have seen a steady increase in the presence of fake accounts (Khaled et al., 2018). These accounts are often used to spread fake news and seed distrust for political gain (Shu et al., 2017). Wikipedia is not immune to such attacks: Saez-Trumper (2019) investigated political and religious groups imposing their narratives on articles. Attacks on Wikipedia are particularly threatening as articles often serve as the ground-truth for automated fact checking systems; used to combat disinformation on other platforms (Thorne et al., 2018).

Changes to articles on Wikipedia are a collaborative process, where decisions are made via the consensus of editors. Malicious users undermine this process through *Sockpuppetry*: the use of multiple accounts to stack votes, fake majority support of a view or make a counter-perspective look absurd (Saez-Trumper, 2019). They have been used to vandalise articles (Solorio et al., 2013a), support political views (Kumar et al., 2017) or improve personal standing (Stone and Richtel, 2007).

Many machine learning approaches have been proposed, including linking sockpuppet accounts through their writing style (Solorio et al., 2013a; Sakib and Spezzano, 2022) using a related technique called *authorship-attribution*. However, when the available text is scarce, it is difficult to profile an author accurately (Eder, 2013). In the case of sockpuppet detection, where available text samples are short (Solorio et al., 2013b), this makes achieving good performance difficult. Previous approaches (Solorio et al., 2013a; Sakib and Spezzano, 2022) manage this challenge by merging the corpus of individual sockpuppet investigations into a single dataset. A model trained on this dataset then learns the writing style of sockpuppets as a whole. Whilst the model can be later fine-tuned, it will not be sensitive to author-specific features.

Meta-learning instead leverages prior experience to perform well on limited data. Rather than merging the corpus of sockpuppet investigations together and training one large model, it considers each as a separate learning task – a new model is trained for each individual investigation. These *base-models* begin as a copy of a *meta-model*, a single model that is updated by what is learned by each base-model. The meta-model learns a general understanding of sockpuppets that can quickly adapt to the behaviour of an unseen *puppetmaster*, the user behind a group of sockpuppets. We elaborate on this process in Section 4.

In this study, we are the first to apply meta-learning to the problem of sockpuppet detection. Our work makes three main contributions:

1. We evaluate the application of meta-learning to the task of sockpuppet detection on

Wikipedia. We find that learning over a distribution of tasks significantly improves prediction precision over pre-trained approaches. This outcome is valuable for sockpuppet detection, where confidence in positive identifications is paramount. Our approach[1] is applicable to other online communities as well, and we release it publicly.

2. We construct and publicly release a dataset of sockpuppet investigations on Wikipedia. Our dataset[1] improves upon existing datasets which are either outdated (Solorio et al., 2013a), unreleased publicly (Kumar et al., 2017) or do not preserve investigation structure (Sakib and Spezzano, 2022).

3. We formulate a more realistic task definition. Previous approaches (Sakib and Spezzano, 2022) train on data from any number of accounts within a sockpuppet-group, preemptively revealing any deceptive efforts made by a puppetmaster to the model. Our model is fine-tuned on just one accused user, as it would be when deployed.

2 Related Work

2.1 Sockpuppetry

Sockpuppetry is typically described as the use of multiple accounts by a single user for deceptive or malicious purposes (Zheng et al., 2011; Solorio et al., 2013a; Bu et al., 2013; Liu et al., 2016; Sakib and Spezzano, 2022), however, not all sockpuppets are malicious. Kumar et al. (2017) provide a more general definition: A *sockpuppet* is any account controlled by a user with at least one other account. The set of these accounts is referred to as a *sockpuppet-group*, and their controlling user their *puppetmaster*. We use this definition with one small amendment: that these accounts be, at some point, operated concurrently. This adjustment distinguishes the task of sockpuppet detection from that of ban evasion, where secondary accounts are created strictly after the primary accounts are banned (Niverthi et al., 2022). Other similar tasks include bot campaign detection, where accounts are controlled by automated agents instead of humans. Advancements in human-like text generation is blurring this distinction (Sallah et al., 2024).

2.1.1 Motivation

Malicious users use sockpuppets to vandalise Wikipedia pages (Solorio et al., 2013a), propagandise political views (Kumar et al., 2017; Afroz et al., 2012), or improve their own public image (Owens, 2013). Sockpuppets undermine collaboration on Wikipedia through false majority opinions, vote stacking (Solorio et al., 2013a) and *Straw man socks*, which argue easily refuted opposing arguments to discredit opposition (Kumar et al., 2017).

2.1.2 Detection

The current approach to detecting sockpuppetry on Wikipedia is manual[2]: users argue their case before a presiding administrator, who may supplement evidence with technical logs. Once a verdict is reached, guilty accounts are suspended and the investigation is archived.

Many automated approaches have been proposed to support this process. *Authorship Attribution* (AA) has been used to determine whether the intent and writing style of accounts are similar enough to be the same user. Linear classifiers with manually selected authorship features have achieved consistent results (Solorio et al., 2013a; Bu et al., 2013; Liu et al., 2016; Sakib and Spezzano, 2022). They use lexical, structural and syntactic features (Bu et al., 2013). AA classifiers struggle when given only short pieces of text (Shrestha et al., 2017), which is typically all that is available in the case of sockpuppet detection (Solorio et al., 2013a). Features may also require domain specific selection (Kotsiantis et al., 2007), and typically act under the assumption that the authors are not attempting to evade detection (Solorio et al., 2013a). Faced with adversarial authors, commonly used authorship features are easily evaded (Brennan et al., 2012).

Meta-data approaches focus on the behaviour of users. Tsikerdekis and Zeadally (2014) identified that the number and time between edits deviates from that of normal users over time. Meta-data approaches typically do not require pairwise comparison between accounts, which reduces computational complexity. Kumar et al. (2017) highlight six points of divergence from usual user activity. These features can be combined with authorship features for improved performance (Solorio et al., 2013a; Sakib and Spezzano, 2022). In all prior literature, approaches consider a single model that classifies users or edits as belonging to a sockpuppet or not, rather than training a new model for each investigation (Solorio et al., 2013a; Bu et al., 2013; Liu et al., 2016; Sakib and Spezzano, 2022).

[1] `https://github.com/lraszewski/wiki-socks`

[2] `https://en.wikipedia.org/wiki/Wikipedia:Sockpuppet_investigations`

2.2 Meta-learning

Meta-learning research focuses on the problem of "learning to learn". In this setting, a machine learning model gains experience over a collection of tasks, rather than just one, and in doing so improves its performance on future tasks (Hospedales et al., 2020). Hospedales et al. (2020) define *base-learning* as the inner learning algorithm solving a task, such as authorship attribution. *Meta-learning* is an outer learning algorithm which updates the inner algorithm according to its own *meta-objective*, typically quick adaptation to new tasks (Finn et al., 2017; Snell et al., 2017; So, 2021).

Meta-learning has been successful in many domains, such as image classification (Antoniou et al., 2019), sentiment analysis (Liang et al., 2023), and text classification (Bansal et al., 2021). Tian et al. (2023) investigated the approach to detect state-sponsored trolls. Beyond reducing data dependence, other limitations of deep neural networks, such as unsupervised learning performance, may also be improved (Hospedales et al., 2020).

Some limitations include a requirement for a large number of tasks (Al-Shedivat et al., 2021), significant compute costs of first-order techniques (Finn et al., 2017), generalisation to out of distribution tasks and a lack of realistic benchmark datasets (Vettoruzzo et al., 2023).

Gradient-based Meta-Learning approaches are made up of several families. Gradient-based approaches use gradient descent to update a model's parameters to minimise the loss according to a meta-objective. These approaches are model-agnostic, making them advantageous over Metric and Model-based approaches that make restrictions on model architecture. The foremost approach is MAML (Finn et al., 2017) and its successors (Antoniou et al., 2019; Triantafillou et al., 2020; Finn et al., 2018; Rajeswaran et al., 2019). A related approach is Reptile (Nichol et al., 2018) that requires significantly less compute whilst achieving similar performance (Vinyals et al., 2016). An advantage of Reptile over MAML is that it does not require a train-test split for each training task (Nichol et al., 2018), allowing more data to be used in training.

Metric-based Metric based approaches learn a distance function (metric) that clusters samples from the same class together. This metric should then generalise well on new tasks (Hospedales et al., 2020), and in most approaches does not require fine-tuning (Sung et al., 2018). They include prototypical networks (Snell et al., 2017), siamese networks (Koch et al., 2015), relation networks (Sung et al., 2018) and matching networks (Vinyals et al., 2016). Unlike gradient-based approaches, they make some restrictions on model-architecture.

Model-based Model-based methods use a model architecture specifically designed for rapid parameter updates. They include extended neural turing machines (Santoro et al., 2016) and meta-networks (Munkhdalai and Yu, 2017). Their main drawback is their inherent restrictions on model design.

3 Dataset

We create and release[3] a novel sockpuppetry dataset for this study. Using the meta-learning paradigm, each investigation is framed as a discrete problem, where writing samples from a single sockpuppet-group must be separated from writing samples of non-sockpuppets.

Tasks consist of writing samples from both sockpuppet and non-sockpuppet users. Edits to Wikipedia are called *contributions*, and contain a *message* component where the user can describe their edit. As in previous approaches (Solorio et al., 2013a; Sakib and Spezzano, 2022), we use these contribution messages as the writing samples. Negative samples are contributions from non-sockpuppet accounts drawn from the same time and article distribution as the sockpuppet-group.

The final dataset consists of $23,610$ tasks. For each contribution, we provide the timestamp, revision ID, ID of the preceding contribution, user name, article title, contribution message, and a binary label. A sample from one investigation is given in Table 6 in Appendix A.

3.1 Data collection

To collect the positive samples, the confirmed Wikipedia sockpuppets page[4] was crawled using a combination of Pywikibot[5] and MediaWiki[6] API calls, which extracted each investigation page and the contributions of each confirmed sockpuppet.

Negative samples for each task were collected from the same articles and within the active time period (first and last contribution) of the task's

[3]https://github.com/lraszewski/wiki-socks
[4]https://en.wikipedia.org/wiki/Category:Wikipedia_sockpuppets
[5]https://github.com/wikimedia/pywikibot
[6]https://www.mediawiki.org/wiki/MediaWiki

sockpuppet-group. For each positive sample, two random timestamps were drawn, and the next ten contributions made after each was collected. From each set of ten, the first valid (non-duplicate, non-sockpuppet) sample was selected. Multiple samples from the set of ten were not collected, so as to maintain a uniformly random temporal distribution. If a set of ten contained no valid negative samples, the attempt was abandoned. This occurred in cases where the active time period was very short, or the articles were inactive or new.

In reality, there is a class imbalance between the number of sockpuppet and non-sockpuppet accounts. Negative contributions were thus over sampled. Arbitrarily, an ideal ratio of two negative samples to each positive was set. It is unclear how an informed estimate might be reached: the true ratio of genuine users to sockpuppets would not only be too extreme to replicate or learn with, but investigations only occur on accused users, not all users. Ideally one might ascertain the ratio of accused sockpuppets to confirmed sockpuppets, however as the investigations of falsely accused accounts are not archived together, this is impractical to obtain.

For investigations that did not reach the ideal ratio of two negatives for each positive, a second identical pass was performed. This strategy over-sampled articles with more non-sockpuppet editors to make up for the shortfall. This yielded some tasks with up to four negatives for each positive.

664 investigations failed to collect any negative samples. This may have occurred for a few reasons, namely that the sockpuppet activity was contained to pages without any non-sockpuppet editors[7], that the sockpuppet activity was confined to a short period of time that did not contain any other editors[8], or that the sockpuppets made very few contributions[9]. These investigations were retained in the dataset, but were excluded from any experiments. 984 investigations contained no positive samples. These were removed, and a list of empty investigations provided alongside the rest of the dataset.

4 Method

We provide two task definitions. The first is a description of the *base-learning* problem, which considers training and evaluating a classifier on a single task. This is also referred to as the *inner-loop* in the context of meta-learning. The second description is of the meta-learning problem, and describes how a meta-model is learnt across a distribution of base-learning tasks. This is also called the *outer-loop*.

4.1 Base-learning

The base-learning task is a binary classification problem. As input, the model will receive two data sources: the article *page* and *message* describing the contribution. The model outputs a classification, identifying the contribution as either a positive (made by a sockpuppet), or negative sample.

In a deployed setting, there is no given list of confirmed sockpuppets, only a set of accused accounts. Therefore, a model may only be trained on the contributions of a single accused user, which may then be tested on the contributions of the remaining accused accounts. We make similar restrictions on our training data: for each investigation, we define the *puppetmaster* as the sockpuppet with the most contributions. A model is then trained on their contributions. This model is assessed by its ability to distinguish the contributions of the remaining sockpuppets from the samples of non-sockpuppets. By contrast, prior works (Sakib and Spezzano, 2022; Solorio et al., 2013a) train on data from a mix of accounts. If a puppetmaster attempts to change their behaviour on sockpuppet accounts, this may be unrealistically revealed to a model in training.

We call the set of puppetmaster samples the train set, and the set of sockpuppet samples the test set. Negative samples are split between the two sets proportionally. Due to how these sets are created, test sets much larger than the train set are common.

A validation set is split from the train set, containing 20% of the available samples and maintaining the same proportion of positive and negative samples. This set is used during base-learning to provide feedback as the model is being trained, and to prevent over-fitting via early stopping. We provide summary statistics of the train-test split in Table 3 in Appendix A.

4.2 Meta-learning

The meta-learning problem considers a distribution of tasks (Finn et al., 2017). This distribution is split into two sets, meta-train and meta-test. The meta-train set is used for the meta-learning process, whilst the meta-test set is used to evaluate how well the meta-learned model performs. The average

[7]https://en.wikipedia.org/wiki/Category:Wikipedia_sockpuppets_of_Alliasalmon

[8]https://en.wikipedia.org/wiki/Category:Wikipedia_sockpuppets_of_Appearedclip

[9]https://en.wikipedia.org/wiki/Category:Wikipedia_sockpuppets_of_Lolawin

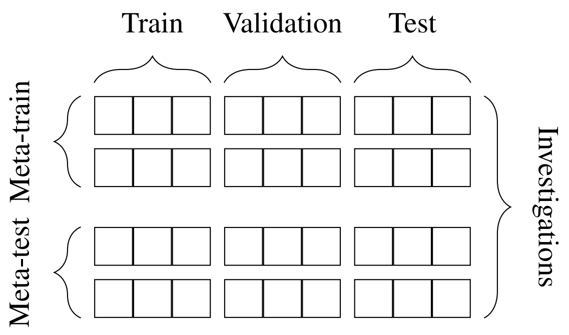

Figure 1: Dataset topology.

performance of the models on these tasks is what is reported in Section 7.

Figure 1 depicts the dataset topology, where each row represents the samples of a task. The tasks are split into the meta-train and meta-test sets, and each task is split into train, validation and test sets.

We make several restrictions on the shape of this distribution. To ensure that each task has an over representation of negative samples, we limit the distribution to only include tasks with a negative to positive ratio of at least one. We also ensure that each task has at least ten puppetmaster samples and five sockpuppet samples. These restrictions are derived from the model architecture, which uses a triplet contrastive loss function that requires at least two positive samples in each task. By ensuring each task contains at least ten puppetmaster samples, we guarantee a valid validation set. These restrictions reduce the total number of tasks from $23,610$ to $13,549$. 90% of tasks ($12,194$) are reserved for the meta-train set, and the rest ($1,355$) become the meta-test set. To compare the model with non-meta-learning approaches, the meta-train set will either be used for the pre-trained approach, or, where no pre-training is necessary, will not be used at all.

Whilst training on the meta-train set of tasks, approaches are not required to maintain the distinction between task specific train, test and validation sets. These sets are only preserved to the extent that they are required for the meta-learning or pre-trained approach. The training process on each meta-test task is kept constant throughout each approach. Each model is given a maximum of ten epochs to train on the new task before predictions must be made. The metrics of each task in the meta-test set are then averaged to create the overall metrics for the approach. Each approach is run three times, and their mean and standard deviation reported in Section 7.

4.3 Meta-learning Strategy

We use Reptile as our meta-learning strategy. This is because it is similarly performant to MAML whilst being less computationally expensive (Nichol et al., 2018; Rajeswaran et al., 2019). Compared to metric and model-based approaches, Reptile has the benefit of being model-agnostic, allowing flexibility in model architecture.

We use the serial implementation that updates the parameters directly through linear interpolation. It works by repeatedly adapting a clone of the meta-model to a task $\mathcal{T}_i$, and then moving the parameters of the original meta-model θ toward the adapted parameters θ' by a factor of ϵ. The parameters of the meta-model therefore move in a direction common to the optimal parameters of each task. The Reptile algorithm is provided in Algorithm 1.

Algorithm 1 Reptile Algorithm (Serial)

```
**Input:** Interpolation rate $\epsilon$, inner steps $k$, task distribution $p(\mathcal{T})$, initial model parameters $\theta$
**for** iteration $= 1, 2, \ldots$ **do**
    Sample a task $\mathcal{T}_i \in p(\mathcal{T})$
    Compute $\theta' = U^k_{\mathcal{T}_i}(\theta)$, denoting $k$ steps of SGD or Adam
    Update $\theta \leftarrow \theta + \epsilon(\theta' - \theta)$
**end for**
```

5 Model Architecture

A diagram of our approach is given in Figure 2. The sample represents a positive or negative contribution. The two textual inputs of the contribution, *page* and *message*, are concatenated using the appropriate separator token. We use RoBERTa[10] (Liu et al., 2019), a pre-trained transformer with frozen parameters, to generate a matrix of contextualised word embeddings. This approach is typical of recent authorship attribution classifiers (De Langhe et al., 2024; Huertas-Tato et al., 2022; Ai et al., 2022; Rivera-Soto et al., 2021).

This matrix is then fed to a transformer encoder, optimised using Adam (Kingma and Ba, 2014). This encoder is where the majority of task learning takes place, and is the model that will be trained using meta-learning.

Once the authorship embeddings are produced, a neural network processes the embedding to produce a logit, which is later transformed to produce

[10] https://huggingface.co/sentence-transformers/all-distilroberta-v1

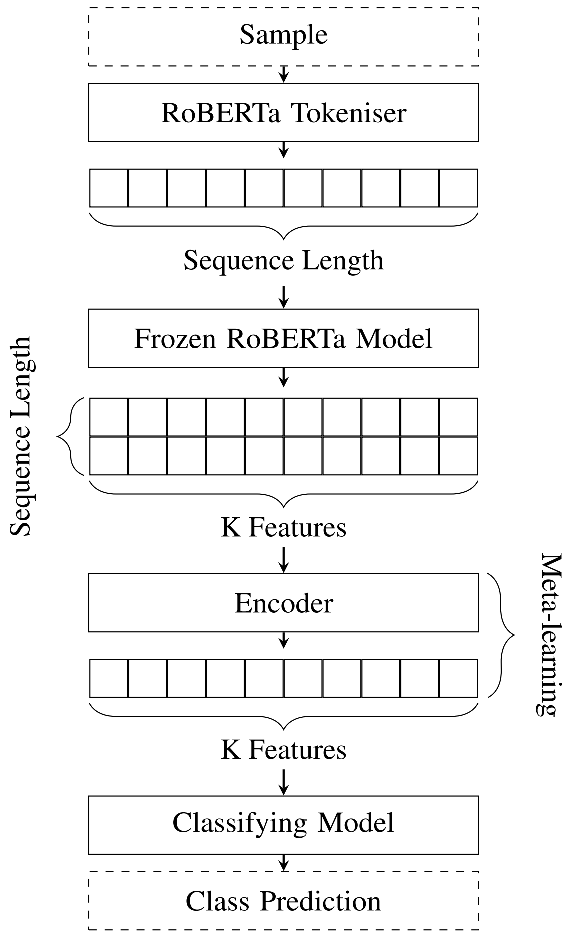

Figure 2: Model topology.

discrete classifications. The classifier has two fully connected layers of dimension 768 with dropout. It also uses the Adam optimiser, and is trained using a cross-entropy loss function. The classifier is trained on the embeddings produced after the encoder has been trained.

5.1 Loss Functions

The encoder model is trained using triplet margin loss (Schroff et al., 2015). A contrastive loss function is typical in natural language tasks comparing document similarity (Pennington et al., 2014; Devlin et al., 2019), and also has prior success in contrastive authorship models (Huertas-Tato et al., 2022). The triplets are created by iterating through all the positive samples as the anchor, and randomly selecting another negative and positive. Typically, the anchor may be drawn from both classes, however we limit triplets to the positive sockpuppet class. This is because the authors in the negative samples are all different, and therefore should occupy different regions in the embedding space – only positive samples should be clustered together. For the classifier models that interpret the embeddings, we use binary cross-entropy loss.

5.2 Training Parameters

When training both the classifier and encoder on the meta-test set of tasks, each was given a maximum of ten epochs, where each epoch is one complete pass through the train set. This is a limitation of the time and computing resources available.

We used a variable batch size strategy that scaled with the length of the task. This catered for smaller tasks whilst still allowing larger tasks to benefit from the stability and speed of larger batch sizes. We used early stopping based on the validation loss with a patience of 3 epochs. During the meta-learning stage, the model was trained over five epochs of the meta-train set of tasks. On each task, Reptile performed five gradient steps before the parameters were updated using an interpolation rate of 0.2. The β_1 parameter of Adam was set to 0 as recommended (Nichol et al., 2018). At the end of each epoch, the model was saved along with the sum of the training loss of each task in that epoch. The best performing model relative to the validation loss was selected.

5.3 Hyper-parameters

We tuned model hyper-parameters using the Optuna[11] framework. The encoder and classifier models were tuned together. 100 tests were run, where the model was trained over ten randomly selected tasks from the meta-train distribution of tasks. The performance of each on their respective tasks was averaged and provided to the optimiser as feedback. We performed three optimisations, one for the encoder and classifier models, a second for the RoBERTa baseline classifier (Section 6.1), and a third for the Reptile parameters. We provide the tuned hyper-parameters in Table 5 in Appendix A.

6 Metrics

The main metrics for comparison between approaches should be the area-based metrics, AUROC and AUPRC. This is because they do not require a specific threshold to be decided, which may distort the appearance of classifier performance. We also provide the F1-Score and F0.5-Score. The first is justified by previous literature (Sakib and Spezzano, 2022; Solorio et al., 2013a,b), whilst the

[11] `https://optuna.org/`

second presents a balance between recall and precision more relevant to the deployment environment, where false positives are strongly discouraged[12]. We also provide the accuracy, precision, and recall of each model as supplementary metrics.

Reported metrics undergo an aggregation process. For each experiment, the results of the approach on the test set of each task in the meta-test set (see Figure 1) are computed. The results of each task are then averaged to find the overall result of the approach for that experiment. Three experiments of each approach are run. The metrics across each experiment are averaged, and the standard deviation provided as a confidence interval.

6.1 Baselines

We consider several baselines, including two trivial baselines (random and majority classifiers), also used by Solorio et al. (2013a). In the case of the majority baseline, the class predicted is based on the training dataset.

RoBERTa baseline To assess whether the encoder model itself provides a significant improvement, we train a simple binary classifier on the sentence-level RoBERTa embeddings. This changes the model architecture by reducing the output from the frozen RoBERTa model from a two-dimensional matrix to a one-dimensional vector. The vector is then fed directly into a fully connected neural network classifier. Huertas-Tato et al. (2022) employed a similar baseline to assess their authorship representation learner.

Non-meta-learning approach To isolate the effects of meta-learning, we also test our approach without it. This model will follow the same training approach as the Meta-learned model on test tasks.

Pre-trained approach The pre-trained approach trains our model on a merged dataset of the meta-train set of tasks. This is the approach used in prior literature (Solorio et al., 2013a; Sakib and Spezzano, 2022). At test time, this approach will be fine-tuned on tasks in the meta-test set.

Upper limit As a significant portion of all contributions do not have any text in their *message* feature (24.45% of all contributions, 63.64% of which are positive samples), these contributions are indistinguishable from one another using the provided features. Therefore, the upper limit of performance is more accurately defined as the perfect classifier on all contributions where a *message* value is present, and a random classifier otherwise.

7 Results

Our results are presented in Table 1. Metrics are measured as the mean across all three test runs with the standard deviation as error bounds. The classification threshold used to compute predictions from logits for applicable metrics were computed at a task level, using the optimal threshold relative to the F0.5-Score on the task validation set. A T-test was used to evaluate the statistical significance of the meta-encoder compared to the standard encoder, indicated with asterisks. The averaged ROC and PR curves are provided in Appendix A.

The meta-learning approach significantly outperforms other approaches ($P << 0.05$) in AUROC, AUPRC, F1-score, F0.5-score and accuracy, with substantial improvements of approximately 10%. Recall did not improve significantly, tying the overall improvement to an increase in precision. This suggests that meta-learning helps the classifier make fewer false positive predictions whilst preserving its ability to identify true positives. This result is desirable for the detection of any malicious behaviour, as unjustly punishing innocent users is typically considered worse than missing a few malicious ones – the emphasis is on a very high confidence in positive predictions.

The metrics of the pre-trained encoder are unexpected. The additional pre-training should have provided the model with a general understanding of the task prior to fine-tuning, however the approach falls behind the non-pre-trained encoder model on most metrics. In prior work (Sakib and Spezzano, 2022; Solorio et al., 2013a), the pre-trained approach performed well. This reduction in performance may be due to the harder task setting, however there may be other causes. Sakib and Spezzano (2022) combined their authorship attribution features with behavioural features, which may be more consistent across tasks, and therefore better for the pre-trained approach. This suggests approaches that focus on authorship attribution may require a model to have greater adaptability.

The lowest performing metric is the AUPRC result. This is likely due to the class imbalance, which overall consisted of 65.60% negative and 34.40% positive samples. The AUPRC is more

[12]https://en.wikipedia.org/w/index.php?title=Wikipedia:Sockpuppet_investigations/SPI/Administrators_instructions&oldid=1173289303

Approach	AUROC	AUPRC	F1-Score	F0.5-Score	Accuracy	Precision	Recall
Random	50.10 ± 0.14	50.85 ± 0.09	40.34 ± 0.11	36.52 ± 0.12	50.10 ± 0.16	34.46 ± 0.12	50.05 ± 0.15
Majority	-	-	-	-	65.60 ± 0.00	-	-
RoBERTa	65.70 ± 0.00	50.45 ± 0.03	57.97 ± 0.01	57.52 ± 0.06	66.98 ± 0.06	59.54 ± 0.13	67.63 ± 0.17
Standard Enc.	68.33 ± 0.09	50.67 ± 0.33	60.05 ± 0.18	58.73 ± 0.16	68.90 ± 0.07	59.72 ± 0.32	69.88 ± 0.34
Pre-trained Enc.	62.74 ± 0.02	44.80 ± 0.19	57.49 ± 0.13	52.90 ± 0.12	62.79 ± 0.05	51.45 ± 0.25	74.76 ± 0.28
Reptile Enc.	**78.98 ± 0.12***	**62.21 ± 0.08***	**67.46 ± 0.53***	**67.89 ± 0.17***	**77.51 ± 0.19***	**69.43 ± 0.26***	**70.81 ± 0.82**
Upper Limit	96.73 ± 0.00	93.56 ± 0.00	86.48 ± 0.00	91.11 ± 0.00	92.01 ± 0.00	95.38 ± 0.00	81.66 ± 0.00

Table 1: Results for the sockpuppet prediction task. Asterisks indicate statistical significance compared to the standard encoder ($p < 0.05$).

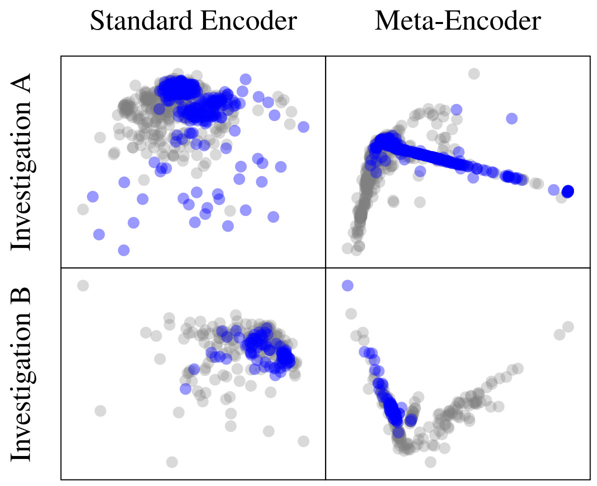

Figure 3: PCA of high performing embeddings. Blue circles represent positive samples.

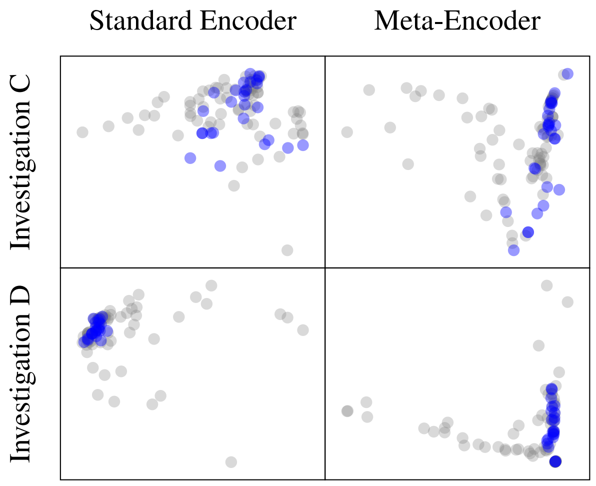

Figure 4: PCA of low performing embeddings. Blue circles represent positive samples.

sensitive to 'hard' negative samples, as precision decreases substantially for each false positive.

This result suggests that despite an increase in the precision of predictions being the principal benefit of meta-learning, it still remains the model's main flaw. This conclusion is further corroborated in Section 7.1. Intriguingly, both non-meta-learning approaches achieved marginally worse scores than the random baseline. The similarly low precision scores corroborate the earlier statement that the principal improvement of meta-learning in this domain is the reduction in false positives.

Surprisingly, the performance difference between the basic RoBERTa classifier and the encoder model is small. The encoder model was expected to perform better as it is trained on the word level embeddings produced by RoBERTa, and therefore should have had a richer understanding of user writing style than the semantic sentence level embeddings used in the RoBERTa classifier. In all metrics the encoder performs slightly better, suggesting there is some truth to the hypothesis, however, the small training set sizes may have prevented a significant divergence.

Whilst prior Wikipedia sockpuppet detection approaches report higher F1-scores of 73 (Solorio et al., 2013a) and 82 (Sakib and Spezzano, 2022), the difference in datasets and task construction (neither study distinguishes between sockpuppets and puppetmasters) make a fair comparison difficult.

7.1 Error Analysis

Figure 3 provides insight into the effect of meta-learning on the embeddings. The embeddings of two test investigations[13] have been projected to two dimensions using Principal Component Analysis (PCA), with positive samples being coloured in blue. The left-hand column contains the embeddings of the test samples produced by the standard encoder model after training on the task. The right-hand column are the embeddings produced with the meta-encoder.

The embeddings learnt by the meta-encoder appear tighter, with less overlap between the clusters. This is supported by the results of these particular investigations: investigation A received an AUROC of 62% with the standard encoder, which improved to 83% using the meta-encoder. Investi-

[13]Investigations of *Film_Fan* (A) and *Al_aman_kollam* (B).

Positive Samples	RoBERTa	Standard Enc.	Pre-trained Enc.	Reptile Enc.
$[10, 20)$	64.66 ± 0.26	66.65 ± 0.55	64.23 ± 0.45	$\mathbf{79.37 \pm 0.99}$*
$[20, 30)$	64.48 ± 0.22	67.32 ± 0.45	64.74 ± 0.33	$\mathbf{80.00 \pm 0.02}$*
$[30, 40)$	63.83 ± 0.31	66.96 ± 0.38	64.56 ± 0.28	$\mathbf{79.01 \pm 0.37}$*
$[40, 50)$	67.81 ± 0.52	70.51 ± 0.41	60.79 ± 0.34	$\mathbf{79.81 \pm 0.22}$*
$[50, \infty)$	66.05 ± 0.04	68.73 ± 0.15	62.20 ± 0.12	$\mathbf{78.60 \pm 0.07}$*

Table 2: AUROC scores of approaches on small tasks. Tasks are binned by the number of positive samples. Asterisks indicate statistical significance compared to the standard encoder ($p < 0.05$).

gation B had a similar improvement, from 69% to 91%. Both standard and meta-encoders were able to cluster positive samples together, however the meta-encoder exhibits better separation from negative samples. This aligns with the overall results, where the meta-encoder saw small improvements in recall, but large improvements in precision.

To contrast the successful examples, Figure 4 presents two investigations[14] that performed poorly. Investigation C achieved AUROCs of 52% (standard encoder) and 54% (meta-encoder). Investigation D was similar, with a small improvement from 58% to 61%. Whilst less defined, the right-angle structure is still evident, and positive samples are still clustered within a single arm, suggesting the encoder has no issues identifying positive samples. The difference then is the proportion of negative samples that appear in the 'positive' arm. This again aligns with the overall results, where recall is largely consistent between approaches, and most of the improvement is in the precision of positive classifications. In these two cases, the poor AUROC performance can be attributed to the failure of the meta-encoder to improve upon the precision.

In Investigation D, contribution messages are characteristically short, typically the name of the article section edited. The messages of many negative samples are similar. This convention is easily detected, explaining why both classifiers were able to cluster the positive samples, but found distinguishing them from negative samples using the convention difficult. This may explain why the recall is acceptable, but precision is low. The sockpuppet-group in investigation C use a message of just a couple of words at most, but typically use no message at all. As most Wikipedians add a contribution message (only 8.89% of negative samples collected had empty message fields, compared to 15.56% of positive samples), a consistently empty one would identify the sockpuppet to the encoder, but would be indistinguishable from legitimate message-less contributions. This leads to the following conclusion: where a sockpuppet's behaviour is characterised by empty or conventional messages, positive recall is strong, but precision suffers.

[14]Investigations of *Amirharbo* (C) and *Cameronfree* (D).

7.2 Small tasks

Table 2 contains additional binned results for each approach on smaller tasks, containing between ten and fifty positive samples each. These results indicate that the performance increase of our model over other approaches tested holds in tasks with exceptionally low samples available.

8 Conclusion

We study the problem of detecting malicious sockpuppetry on Wikipedia. We are the first to propose meta-learning to address the data-scarcity challenge in detecting sockpuppet accounts through writing style. Our results demonstrate significant performance improvements when compared to pre-trained approaches, especially in prediction precision. We attribute this to our approach's ability to quickly adapt to distinct authorship styles with limited samples. In doing so, we defined a more realistic task definition that provides a more accurate measure of performance, and released an updated, verifiable and adaptable dataset of sockpuppet investigations appropriate for future meta-learning research. Our findings extend to any online social platform where users engage in sockpuppetry.

Limitations

There are several limitations of our model that could benefit from further research.

As discussed in Section 7.1, our model is limited in cases where sockpuppet contributions contain little or no message data. In these cases, the encoder requires additional information. One way to address this would be to include the edit data of contributions, that is the changes made to the article itself. This would allow a model to understand the intent and implications of a contribution even when a description is absent. A contribution must make changes to the article, and edits themselves are likely to be far more diverse in nature than the messages, providing a model with a strong distinguishing signal. The additional signal would also further improve high performing investigations.

Another limitation is in safety. There are several legitimate reasons why a user might have several accounts. One of these reasons may be for the safety of editors editing politically contentious articles[15]. Whilst our approach was trained solely on malicious examples of sockpuppetry, no efforts were made to ensure this approach could not reveal benign sockpuppet-groups by mistake. Additional work may focus on providing a safeguard measure that ensures the sockpuppet behaviour being observed is malicious.

Our approach should also be evaluated against Generative language models, which are becoming increasingly effective at creating text that looks human (Liu et al., 2023). It is likely that future sockpuppet-groups might utilise generative models to edit Wikipedia, rewriting edits to conceal writing style or to fully automate contributions. Many approaches are already focusing on the detection of text generated by prolific models (Dhaini et al., 2023). Future work may evaluate how robust the meta-learning approach is to authorship obfuscation using generative models.

Considering the performance of the approach, it is unable to replace human-led sockpuppet investigations. When found to be guilty of sockpuppetry, accounts are blocked. Some users assign great value to their accounts, and incorrect sockpuppet classifications would be damaging to the community. The approach could serve as an additional source of evidence in open investigations, or as a detection method that triggers human-led investigation on suspicious accounts. To occupy a larger role in investigations, the precision of the approach must improve further.

Ethical Considerations

There is a valid concern for privacy when releasing this dataset. Usernames are important to Wikipedia editors (Asikin-Garmager et al., 2025), and may be used to represent a person's real identity, contain some personally identifiable information, or obscure their identity completely.

[15]https://en.wikipedia.org/wiki/List_of_people_imprisoned_for_editing_Wikipedia

Arguments against anonymisation are numerous. Whilst this study does not use username data, previous approaches have (Sakib and Spezzano, 2022), and future approaches may too. As all the data collected is publicly available, any anonymisation attempts would be circumventable. We commit to removing any user who requests removal from our dataset, but maintain the username is a valuable data point for future work. For these reasons, the data was not anonymised.

References

Sadia Afroz, Michael Brennan, and Rachel Greenstadt. 2012. Detecting hoaxes, frauds, and deception in writing style online. In *2012 IEEE Symposium on Security and Privacy*, pages 461–475.

Bo Ai, Yuchen Wang, Yugin Tan, and Samson Tan. 2022. Whodunit? learning to contrast for authorship attribution. In *Proceedings of the 2nd Conference of the Asia-Pacific Chapter of the Association for Computational Linguistics and the 12th International Joint Conference on Natural Language Processing*.

Maruan Al-Shedivat, Liam Li, Eric Xing, and Ameet Talwalkar. 2021. On data efficiency of meta-learning. In *Proceedings of The 24th International Conference on Artificial Intelligence and Statistics*, pages 1369–1377.

Antreas Antoniou, Harrison Edwards, and Amos Storkey. 2019. How to train your maml. In *Seventh International Conference on Learning Representations*.

Eli Asikin-Garmager, Michael Zimmer, Cameran Ashraf, and Leila Zia. 2025. Research and privacy on wikipedia.

Trapit Bansal, Karthick Gunasekaran, Tong Wang, Tsendsuren Munkhdalai, and Andrew McCallum. 2021. Diverse distributions of self-supervised tasks for meta-learning in nlp. In *EMNLP*, pages 5812–5824.

Michael Brennan, Sadia Afroz, and Rachel Greenstadt. 2012. Adversarial stylometry: Circumventing authorship recognition to preserve privacy and anonymity. *ACM Trans. Inf. Syst. Secur.*, pages 12:1–12:22.

Zhan Bu, Zhengyou Xia, and Jiandong Wang. 2013. A sock puppet detection algorithm on virtual spaces. *Knowledge-Based Systems*, pages 366–377.

Loic De Langhe, Orphee De Clercq, and Veronique Hoste. 2024. Unsupervised authorship attribution for medieval Latin using transformer-based embeddings. In *Proceedings of the Third Workshop on Language Technologies for Historical and Ancient Languages (LT4HALA) @ LREC-COLING-2024*, pages 57–64.

Jacob Devlin, Ming-Wei Chang, Kenton Lee, and Kristina Toutanova. 2019. Bert: Pre-training of deep bidirectional transformers for language understanding. In *Proceedings of the 2019 Conference of the North American Chapter of the Association for Computational Linguistics: Human Language Technologies*, pages 4171–4186.

Mahdi Dhaini, Wessel Poelman, and Ege Erdogan. 2023. Detecting ChatGPT: A survey of the state of detecting ChatGPT-generated text. In *Proceedings of the 8th Student Research Workshop associated with the International Conference Recent Advances in Natural Language Processing*, pages 1–12.

Maciej Eder. 2013. Does size matter? authorship attribution, small samples, big problem. *Digital Scholarship in the Humanities*, pages 167–182.

Chelsea Finn, Pieter Abbeel, and Sergey Levine. 2017. Model-agnostic meta-learning for fast adaptation of deep networks. In *Proceedings of the 34th International Conference on Machine Learning - Volume 70*, page 1126–1135.

Chelsea Finn, Kelvin Xu, and Sergey Levine. 2018. Probabilistic model-agnostic meta-learning. In *NeurIPS*.

Timothy Hospedales, Antreas Antoniou, Paul Micaelli, and Amos Storkey. 2020. Meta-learning in neural networks: A survey. *IEEE Transactions on Pattern Analysis and Machine Intelligence*, pages 5149–5169.

Javier Huertas-Tato, Alvaro Huertas-Garcia, Alejandro Martin, and David Camacho. 2022. Part: Pre-trained authorship representation transformer.

Sarah Khaled, Neamat El-Tazi, and Hoda M. O. Mokhtar. 2018. Detecting fake accounts on social media. In *2018 IEEE International Conference on Big Data*.

Diederik Kingma and Jimmy Ba. 2014. Adam: A method for stochastic optimization. *International Conference on Learning Representations*.

Gregory Koch, Richard Zemel, and Ruslan Salakhutdinov. 2015. Siamese neural networks for one-shot image recognition. In *ICML Deep Learning Workshop*.

Sotiris B Kotsiantis, Ioannis Zaharakis, P Pintelas, et al. 2007. Supervised machine learning: A review of classification techniques. *Emerging artificial intelligence applications in computer engineering*, pages 3–24.

Srijan Kumar, Justin Cheng, Jure Leskovec, and V.S. Subrahmanian. 2017. An army of me: Sockpuppets in online discussion communities. In *Proceedings of the 26th International Conference on World Wide Web*, page 857–866.

Bin Liang, Xiang Li, Lin Gui, Yonghao Fu, Yulan He, Min Yang, and Ruifeng Xu. 2023. Few-shot aspect category sentiment analysis via meta-learning. *ACM Trans. Inf. Syst.*, pages 22:1–22:31.

Dong Liu, Quanyuan Wu, Weihong Han, and Bin Zhou. 2016. Sockpuppet gang detection on social media sites. *Frontiers of Computer Science*, pages 124–135.

Yiheng Liu, Tianle Han, Siyuan Ma, Jiayue Zhang, Yuanyuan Yang, Jiaming Tian, Hao He, Antong Li, Mengshen He, Zhengliang Liu, Zihao Wu, Lin Zhao, Dajiang Zhu, Xiang Li, Ning Qiang, Dingang Shen, Tianming Liu, and Bao Ge. 2023. Summary of chatgpt-related research and perspective towards the future of large language models. *Meta-Radiology*.

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.

Tsendsuren Munkhdalai and Hong Yu. 2017. Meta networks. In *ICML*.

Alex Nichol, Joshua Achiam, and John Schulman. 2018. On first-order meta-learning algorithms.

Manoj Niverthi, Gaurav Verma, and Srijan Kumar. 2022. Characterizing, detecting, and predicting online ban evasion. In *Proceedings of the ACM Web Conference 2022*, page 2614–2623.

Simon Owens. 2013. The battle to destroy wikipedia's biggest sockpuppet army.

Jeffrey Pennington, Richard Socher, and Christopher Manning. 2014. GloVe: Global vectors for word representation. In *EMNLP*, pages 1532–1543.

Aravind Rajeswaran, Chelsea Finn, Sham M Kakade, and Sergey Levine. 2019. Meta-learning with implicit gradients. In *NeurIPS*.

Rafael A. Rivera-Soto, Olivia Elizabeth Miano, Juanita Ordonez, Barry Y. Chen, Aleem Khan, Marcus Bishop, and Nicholas Andrews. 2021. Learning universal authorship representations. In *EMNLP*.

Diego Saez-Trumper. 2019. Online disinformation and the role of wikipedia.

Mostofa Najmus Sakib and Francesca Spezzano. 2022. Automated detection of sockpuppet accounts in wikipedia. In *2022 IEEE/ACM International Conference on Advances in Social Networks Analysis and Mining (ASONAM)*, pages 155–158.

Amine Sallah, El Arbi Abdellaoui Alaoui, Said Agoujil, Mudasir Ahmad Wani, Mohamed Hammad, Yassine Maleh, and Ahmed A. Abd El-Latif. 2024. Fine-tuned understanding: Enhancing social bot detection with transformer-based classification. *IEEE Access*, pages 118250–118269.

Adam Santoro, Sergey Bartunov, Matthew Botvinick, Daan Wierstra, and Timothy Lillicrap. 2016. Meta-learning with memory-augmented neural networks. In *Proceedings of the 33rd International Conference on International Conference on Machine Learning - Volume 48*, page 1842–1850.

Florian Schroff, Dmitry Kalenichenko, and James Philbin. 2015. Facenet: A unified embedding for face recognition and clustering. In *2015 IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*, pages 815—-823.

Prasha Shrestha, Sebastian Sierra, Fabio González, Manuel Montes, Paolo Rosso, and Thamar Solorio. 2017. Convolutional neural networks for authorship attribution of short texts. In *EACL*, pages 669–674.

Kai Shu, Amy Sliva, Suhang Wang, Jiliang Tang, and Huan Liu. 2017. Fake news detection on social media: A data mining perspective. *SIGKDD Explor. Newsl.*

Jake Snell, Kevin Swersky, and Richard S. Zemel. 2017. Prototypical networks for few-shot learning. In *NeurIPS*, page 4080–4090.

Chaehan So. 2021. Exploring meta learning: Parameterizing the learning-to-learn process for image classification. In *2021 International Conference on Artificial Intelligence in Information and Communication (ICAIIC)*, pages 199–202.

Thamar Solorio, Ragib Hasan, and Mainul Mizan. 2013a. A case study of sockpuppet detection in wikipedia. In *Proceedings of the Workshop on Language Analysis in Social Media*.

Thamar Solorio, Ragib Hasan, and Mainul Mizan. 2013b. Sockpuppet detection in wikipedia: A corpus of real-world deceptive writing for linking identities. In *International Conference on Language Resources and Evaluation*.

Brad Stone and Matt Richtel. 2007. The hand that controls the sock puppet could get slapped. *The New York Times*.

Flood Sung, Yongxin Yang, Li Zhang, Tao Xiang, Philip H.S. Torr, and Timothy M. Hospedales. 2018. Learning to compare: Relation network for few-shot learning. In *2018 IEEE/CVF Conference on Computer Vision and Pattern Recognition*.

James Thorne, Andreas Vlachos, Christos Christodoulopoulos, and Arpit Mittal. 2018. FEVER: a large-scale dataset for fact extraction and VERification. In *NAACL-HLT*.

Lin Tian, Xiuzhen Zhang, and Jey Han Lau. 2023. Metatroll: Few-shot detection of state-sponsored trolls with transformer adapters. In *Proceedings of the ACM Web Conference 2023*.

Eleni Triantafillou, Tyler Zhu, Vincent Dumoulin, Pascal Lamblin, Utku Evci, Kelvin Xu, Ross Goroshin, Carles Gelada, Kevin Swersky, Pierre-Antoine Manzagol, and Hugo Larochelle. 2020. Meta-dataset: A dataset of datasets for learning to learn from few examples.

Michail Tsikerdekis and Sherali Zeadally. 2014. Multiple account identity deception detection in social media using nonverbal behavior. *IEEE Transactions on Information Forensics and Security*, pages 1311–1321.

Anna Vettoruzzo, Mohamed-Rafik Bouguelia, Joaquin Vanschoren, Thorsteinn Rögnvaldsson, and KC Santosh. 2023. Advances and challenges in meta-learning: A technical review. *IEEE Transactions on Pattern Analysis and Machine Intelligence*, pages 1–20.

Oriol Vinyals, Charles Blundell, Timothy Lillicrap, koray kavukcuoglu, and Daan Wierstra. 2016. Matching networks for one shot learning. In *NeurIPS*.

Xueling Zheng, Yiu Ming Lai, K.P. Chow, Lucas C.K. Hui, and S.M. Yiu. 2011. Sockpuppet detection in online discussion forums. In *2011 Seventh International Conference on Intelligent Information Hiding and Multimedia Signal Processing*.

A Appendix

Statistic	**Value**
Meta-Train Size	12194
Meta-Test Size	1355
Ave. Train Size	534.35
Ave. Validation Size	133.58
Ave. Test Size	923.75
Ave. Train Positives	174.71
Ave. Train Negatives	359.63
Ave. Validation Positives	43.67
Ave. Validation Negatives	89.91
Ave. Test Positives	298.74
Ave. Test Negatives	625.01

Table 3: Train-Test Split statistics.

Statistic	**Value**
Num. Investigations	23160
Ave. Length (Contributions)	970.39
Ave. Positive Samples	321.47
Ave. Negative Samples	648.91
Ave. Puppetmaster Samples	144.27
Ave. Sockpuppet Samples	177.20
Ave. Message Length (Char)	44.92

Table 4: Summary Statistics of our dataset.

Model	Hyper-parameter	Value
Encoder Model	Number of Attention Heads	2
	Number of Layers	6
	Learning Rate	0.0001
	Loss Margin	0.2
	Optimiser	Adam
Classification Model	Dropout Chance	0.35
	Learning Rate	0.001
	Layer 0 Nodes	768
	Layer 1 Nodes	768
	Optimiser	Adam
RoBERTa Classifier	Dropout Chance	0.7615
	Learning Rate	0.0008
	Layer 0 Nodes	768
	Layer 1 Nodes	512
	Layer 2 Nodes	512
	Layer 3 Nodes	256
	Layer 4 Nodes	256
	Layer 5 Nodes	128
	Optimiser	Adam
Reptile	Interpolation Rate	0.2
	Number of Steps	5

Table 5: Tuned model hyper-parameters.

timestamp	revid	parentid	sock	user	page	message
2022-03-15T23:45:35+00:00	1077368891	1077368777	1	user1	IShowSpeed	fixed errors
2022-03-15T23:44:47+00:00	1077368777	1077368713	1	user2	IShowSpeed	Added Michigan
2022-04-28T20:45:59+00:00	1085166027	1085165878	1	user3	Cyclone Batsirai	Where's the source??
2020-10-01T00:44:54+00:00	981220197	981144214	0	user4	User talk:Ohnoitsjamie	/* William Stickman IV partisan editing */
2019-04-22T01:46:29+00:00	893534911	892677927	0	user5	User talk:TheresNoTime	/* Come back! */ new section

Table 6: Data sample from one investigation.

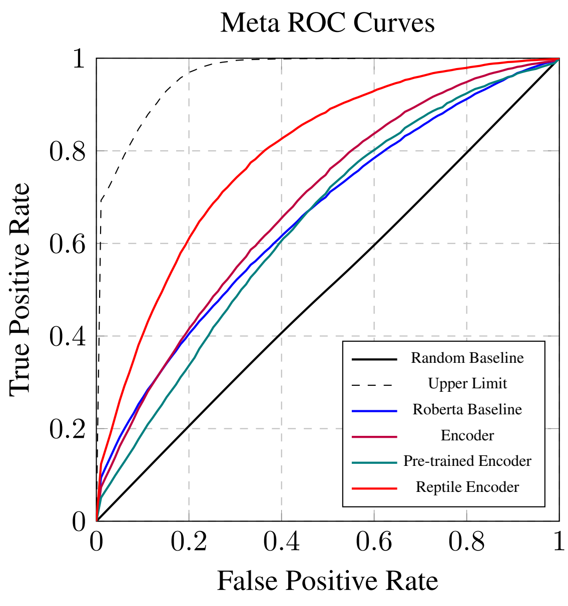

Figure 5: Average ROC curves of models on test tasks.

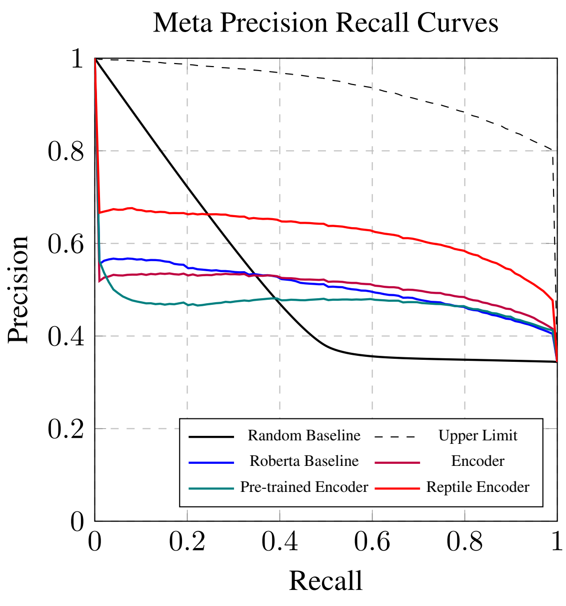

Figure 6: Average PR curves of models on test tasks.