FIESTA: Fast IdEntification of State-of-The-Art models using adaptive bandit algorithms

Henry B. Moss
STOR-i Centre for
Doctoral Training,
Lancaster University

Andrew Moore
School of Computing
and Communications,
Lancaster University

David S. Leslie
Dept. of Mathematics
and Statistics,
Lancaster University

Paul Rayson
School of Computing
and Communications,
Lancaster University

initial.surname@lancaster.ac.uk

Abstract

We present FIESTA, a model selection approach that significantly reduces the computational resources required to reliably identify *state-of-the-art* performance from large collections of candidate models. Despite being known to produce unreliable comparisons, it is still common practice to compare model evaluations based on single choices of random seeds. We show that reliable model selection also requires evaluations based on multiple train-test splits (contrary to common practice in many shared tasks). Using bandit theory from the statistics literature, we are able to adaptively determine appropriate numbers of data splits and random seeds used to evaluate each model, focusing computational resources on the evaluation of promising models whilst avoiding wasting evaluations on models with lower performance. Furthermore, our user-friendly Python implementation produces confidence guarantees of correctly selecting the optimal model. We evaluate our algorithms by selecting between 8 target-dependent sentiment analysis methods using dramatically fewer model evaluations than current model selection approaches.

1 Introduction and Background

Natural Language Processing (NLP) is a field driven by empirical evaluations. Authors are under pressure to demonstrate that their models or methods achieve *state-of-the-art* performance on a particular task or dataset, which by definition requires reliable model comparison. As models become more numerous, require larger computational resources to train, and the performance of competing models gets closer, the task of reliable model selection has not only become more important, but also increasingly difficult. Without full disclosure of model settings and data splits, it is impossible to accurately compare methods and models.

To be able to perform meaningful model comparisons, we need to be able to reliably evaluate models. Unfortunately, evaluating a model is a non-trivial task and the best we can do is to produce noisy estimates of model performance with the following two distinct sources of stochasticity:

1. We only have access to a finite training dataset, however, evaluating a model on its training data leads to severe over-estimates of performance. To evaluate models without over-fitting, practitioners typically randomly partitioning data into independent training and testing sets, producing estimates that are random quantities with often high variability for NLP problems (Moss et al., 2018). Although methods like bootstrapping (Efron and Tibshirani, 1994) and leave-one-out cross validation (Kohavi, 1995) can provide deterministic estimates of performance, they require the fitting of a large number of models and so are not computationally feasible for the complex models and large data prevalent in NLP. Standard NLP model evaluation strategies range from using a simple (and computationally cheap) single train-test split, to the more sophisticated K-fold cross validation, CV (Kohavi, 1995).

2. The vast majority of recent NLP models are non-deterministic and so their performance has another source of stochasticity, controlled by the choice of random seed during training. Common sources of model instability in modern NLP include weight initialisation, data sub-sampling for stochastic gradient calculation, negative sampling used to train word embeddings (Mikolov et al., 2013) and feature sub-sampling for ensemble methods. In particular, the often *state-of-the-art* LSTMs (and its many variants) have been

Proceedings of the 57th Annual Meeting of the Association for Computational Linguistics, pages 2920–2930
Florence, Italy, July 28 - August 2, 2019. ©2019 Association for Computational Linguistics

shown to exhibit high sensitivity to random seeds (Reimers and Gurevych, 2017).

For reliable model selection, it is crucial to take into account both sources of variability when estimating model performance. Observing a higher score for one model could be a consequence of a particularly non-representative train-test split and/or random seed used to evaluate the model rather than a genuine model improvement. This subtlety is ignored by large scale NLP competitions such as SemEval with evaluations based on a pre-determined train-test split.

Although more precise model evaluations can be obtained with higher computation, calculating overly precise model evaluations is a huge waste of computational resource. On the other hand, our evaluations need to provide reliable conclusions (with only a small probability of selecting a sub-optimal model). It is poorly understood how to choose an appropriate evaluation strategy for a given model selection problem. These are task specific, depending on model stability, the closeness in performance of competing models and subtle properties of the data such as the representativeness of train-test splits.

In contrast to common practice, we consider model selection as a sequential process. Rather than using a fixed evaluation strategy for each model (which we refer to as a non-adaptive approach), we start with a cheap evaluation of each model on just a single train-test split, and then cleverly choose where to allocate further computational resources based on the observed evaluations. If we decide to further test a promising model, we calculate an additional evaluation based on another data split and seed, observing both sources of evaluation variability and allowing reliable assessments of performance.

To perform sequential model fitting, we borrow methods from the multi-armed-bandit (MAB) statistical literature (Lai and Robbins, 1985). This field covers problems motivated by designing optimal strategies for pulling the arms of a bandit (also known as a slot machine) in casinos. Each arm produces rewards from different random distributions which the user must learn by pulling arms. In particular, model selection is equivalent to the problem of best-arm-identification; identifying the arm with the highest mean. Although appearing simple at a first glance, this problem is deceptively complex and has provided motivation for efficient algorithms in a wide range of domains, including clinical trials (Villar et al., 2015) and recommendation systems (Li et al., 2010).

Although we believe that we are the first to use bandits to reduce the cost and improve the reliability of model selection, we are not the first to use them in NLP. Recent work in machine translation makes use of another major part of the MAB literature, seeking to optimise the long-term performance of translation algorithms (Nguyen et al., 2017; Sokolov et al., 2016; Lawrence et al., 2017). Within NLP, our work is most similar to Haffari et al. (2017), who use bandits to minimise the number of data queries required to calculate the F-scores of models. However, this work does not consider the stochasticity of the resulting estimates or easily extend to other evaluation metrics.

The main contribution of this paper is the application of three intuitive algorithms to model selection in NLP, alongside a user-friendly Python implementation: FIESTA (Fast IdEntification of State-of-The-Art)[1]. We can automatically identify an optimal model from large collections of candidate models to a user-chosen confidence level in a small number of model evaluations. We focus on three distinct scenarios that are of interest to the NLP community. Firstly, we consider the **fixed budget** (FB) model selection problem (Section 4.1), a situation common in industry, where a fixed quota of computational resources (or time constraints for real-time decisions) must be appropriately allocated to identify an optimal model with the highest possible confidence. In contrast, we also consider the **fixed confidence** (FC) problem (Section 4.2), which we expect to be of more use for researchers. Here, we wish to claim with a specified confidence level that our selected model is state-of-the-art against a collection of competing models using the minimal amount of computation. Finally, we also consider an extension to the FC scenario, where a practitioner has the computational capacity to fit multiple models in parallel. We demonstrate the effectiveness of our procedures over current model selection approaches when identifying an optimal target-dependent sentiment analysis model from a set of eight competing candidate models (Section 5).

[1] `https://github.com/apmoore1/fiesta`

2 Motivating example

We now provide evidence for the need to vary both data splits and random seeds for reliable model selection. We extend the motivating example used in the work of Reimers and Gurevych (2017), comparing two LSTM-based Named Entity Recognition (NER) models by Ma and Hovy (2016) and Lample et al. (2016), differing only in character representation (via a CNN and a LSTM respectively). We base model training on Ma and Hovy (2016), however, following the settings of Yang et al. (2018) we use a batch size of 64, a weight decay of $10e^{-9}$ and removed momentum. We ran each of the NER models five times with a different random seed on 150 different train, validation, and test splits[2]. Reimers and Gurevych (2017) showed the effect of model instability between these two models, where changing the model's random seeds can lead to drawing different conclusions about which model performed best. We extend this argument by showing that different conclusions can also be drawn if we instead vary the train-test split used for the model evaluation (Figure 1). We see that while data splits 0 and 2 correctly suggest that the LSTM is optimal, using data split 1 suggests the opposite. Therefore, it is clear that we must vary both the random seeds and train-test splits used to evaluate our models if we want reliable model selection.

3 Problem Statement

Extending notation from Arcuri and Briand (2014), we can precisely state the task of selecting between a collection of N candidate models $S = \{m_1, m_2, ..m_N\}$ as finding

$$m^* = \underset{m \in S}{\operatorname{argmax}} \mathcal{M}(m). \tag{1}$$

m^* is the best model according to some chosen evaluation metric $\mathcal{M}$ that measures the performance of that model, e.g accuracy, F-score or AUC (for an summary of model evaluation metrics see Friedman et al. (2001)).

As already argued, Equation (1) paints an overly simplistic picture of model selection. In reality we only have access to noisy realisations of the true model score $\mathcal{M}(m)$ and direct comparisons of single realisations of random variables are unreliable. Therefore, we follow the arguments of Reimers and Gurevych (2018) and consider a meaningful way of comparing noisy model evaluations: namely, finding the model with largest expected performance estimate across different train-test splits and random seeds. Defining the mean performance of model m as μ_m, we see that the task of model selection is equivalent to the accurate learning and comparison of these N unknown means:

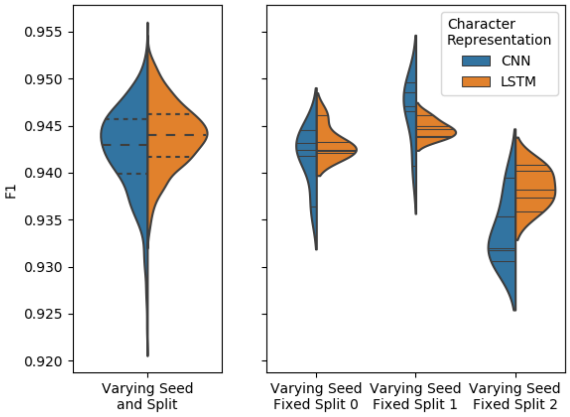

Figure 1: The left plot shows the distribution of results when varying the data splits and random seeds, with the dashed lines representing the quartile values. The three right plots each represent a different single data split over five runs on different random seeds. The lines represent a single run result.

$$m^* = \underset{m \in S}{\operatorname{argmax}} \mu_m.$$

We can now set up the sequential framework of our model selection procedure and precisely state what we mean by reliable model selection. At each step in our algorithm we choose a model to evaluate and sample a performance estimate by randomly generating a data split and random seed. After collecting evaluations, we can calculate sample means for each model, which we denote as $\hat{\mu}_m$. After running our algorithm for T steps, reliable model selection corresponds to knowing how confident we should be that our chosen model $\hat{m}_T = \operatorname{argmax} \hat{\mu}_m$ is in fact the true optimal model m^*, i.e. we wish to make a precise statement of the form;

$$\mathbb{P}\left(\hat{m}_T = m^*\right) \geq 1 - \delta, \tag{2}$$

where $1 - \delta$ represents this confidence.

[2] The original CoNLL data was split with respect to time rather than random sub-sampling, explaining the discrepancy with previous scores on this dataset using the same models.

In Section 1 we motivated two distinct goals of a sequential model selection routine, which we can now state as:

1. **Fixed budget model selection** (FB): We wish to find the best model using only a fixed budget of T model evaluations. The aim is to collect the T evaluations that allow us to claim (2) with the largest possible confidence level $1 - \delta$.

2. **Fixed confidence model selection** (FC): We wish to find the best model to a pre-specified confidence level. The aim is to collect the minimal number of model evaluations that allow us to claim (2).

Although an algorithm designed to do well in one of these scenarios will likely also do well in the other, we will see that to achieve the best performance at either FB or FC model selection, we require subtly different algorithms.

4 Algorithms

We now examine model selection from a bandit viewpoint, summarising three bandit algorithms and relating their use to three distinct model selection scenarios. Although the underpinning theoretical arguments for these algorithms are beyond the scope of this work, we do highlight one point that is relevant for model selection; that scenarios enjoying the largest efficiency gains from moving to adaptive algorithms are those where only a subset of arms have performance close to optimal (Jamieson et al., 2013). Model selection in NLP is often in this scenario, with only a small number of considered models being close to *state-of-the-art*, and so (as we demonstrate in Section 5) NLP has a lot to gain from using our adaptive model selection algorithms.

4.1 Fixed Budget by Sequential Halving

FB best-arm identification algorithms are typically based on successively eliminating arms until just a single (ideally) optimal arm remains (Jamieson et al., 2013; Jamieson and Nowak, 2014; Audibert and Bubeck, 2010). We focus on the **sequential halving** (SH) algorithm of Karnin et al. (2013) (Algorithm 1). Here we break our model selection routine into a series of $\lfloor log_2 N \rfloor$ rounds, each discarding the least promising half of our candidate model set, eventually resulting in a single remaining model. Our computational budget T is split equally among the rounds to be equally budgeted among the models remaining in that round. This allocation strategy ensures an efficient use of resources, for example the surviving final two models are evaluated $2^{\lfloor log_2 N \rfloor} - 1$ times as often as the models eliminated in the first round. An example run of the algorithm is summarised in Table 1.

Algorithm 1 Sequential Halving for Fixed Budget Model Selection

```
Require: Computational Budget $T$,
         Set of $N$ candidate models $S$
  while $|S| \neq 1$ do
    Evaluate each model $m$ in $S$ $\left\lfloor \frac{T}{|S|\lceil \log_2 N \rceil} \right\rfloor$ times
    Update the empirical means $\hat{\mu}_m$
    Remove $\left\lfloor \frac{|S|}{2} \right\rfloor$ models with worst $\hat{\mu}_m$ from $S$
  end while
  return Chosen model $S$
```

Round	Candidate Models	# Evaluations
1	$S = \{m_1, m_2, m_3, m_4\}$	2
2	$S = \{m_2, m_4\}$	4
output:	$S = \{m_2\}$	

Table 1: An example of sequential elimination selecting between four models with a budget of $T = 16$. After two evaluations of each model, two models are eliminated. The remaining budget is then used to reliably decide between the remaining pair. Standard practice would evaluate each model an equal four times, wasting computational resources on sub-optimal models.

In the bandit literature (Karnin et al., 2013), this algorithm is shown to have strong theoretical guarantees of reliably choosing the optimal arm, as long as the reward-distributions for each arm are bounded (limited to some finite range). This is not a restrictive assumption for NLP, as the majority of common performance metrics are bounded, for example accuracy, recall, precision and F-score are all constrained to lie in $[0, 1]$. We will demonstrate the effectiveness of sequential halving for model selection in Section 5.

4.2 Fixed Confidence by TTTS

For fixed confidence model selection, where we wish to guarantee the selection of an optimal arm at a given confidence level, we cannot just discard arms that are likely to be sub-optimal without accurately estimating this likelihood of sub-optimality. Although approaches that sequentially eliminate arms (like our sequential halving algorithm) do exist for FC best-arm identification (Jamieson et al., 2014; Karnin et al., 2013; Audibert and Bubeck, 2010; Even-Dar et al., 2002), the best theoretical guarantees for the FC problem come from algorithms that maintain the ability to sample any arm at any point in the selection procedure (Garivier and Kaufmann, 2016; Jamieson and Nowak, 2014). Rather than seeking to eliminate half the considered models at regular intervals of computation, a model is only evaluated until we can be sufficiently confident that it is sub-optimal. Unfortunately, the performance guarantees for these methods are asymptotic results (in the number of arms and the number of arm pulls) and have little practical relevance to the (at most) tens of arms in a model selection problem.

Our practical recommendation for FC model selection is a variant of the well-known Bayesian sampling algorithm, Thompson sampling, known as **top-two Thompson sampling** (TTTS) (Russo, 2016). We will see that this algorithm can efficiently allocate computational resources to quickly find optimal models. Furthermore, this approach provides full uncertainty estimation over the final choice of model, providing the confidence guarantees required for FC model selection.

Our implementation makes the assumption that the evaluations of each model roughly follow a Gaussian distribution, with different means and variances. Although such assumptions are common in the model evaluation literature (Reimers and Gurevych, 2018) and for statistical testing in NLP (Dror et al., 2018), they could be problematic for the bounded metrics common in NLP. Therefore we also experimented with modelling the logit transformation of our evaluations, mapping our evaluation metric to the whole real line. However, for our examples of Section 5 we found that this mapping provided a negligible improvement in reliability and so was not worth including in our experimental results. This may not be the case for other tasks or less well-behaved evaluation metrics and so we include this functionality in the FIESTA package.

Algorithm 2 Top-Two Thompson Sampling

```
Require: Desired Confidence $1-\delta$,
         Set of $N$ candidate models $S$
  Initialise a uniform belief $\pi$
  Evaluate each model in $S$ three times [3]
  Update belief $\pi$
  while $\max_{m\in S}\pi_m \leq 1-\delta$ do
    Sample distinct $m_1$ and $m_2$ according to $\pi$
    Randomly choose between $m_1$ and $m_2$
    Evaluate chosen model
    Update belief $\pi$
  end while
  return Chosen model $\operatorname{argmax}_{m\in S}\pi_m$
```

To provide efficient model selection, we use our current believed probability that a given model is optimal $\pi_m = \mathbb{P}(m^* = m)$ (producing a distribution over the models $\pi = \{\pi_1, .., \pi_N\}$) to drive the allocation of computational resources. Standard Thompson sampling is a stochastic algorithm that generates a choice of model by sampling from our current belief π, i.e. choosing to evaluate a model with the same probability that we believe is optimal (see Russo et al. (2018) for a concise introduction). Although this strategy allows us to focus computation on promising arms, it actually does so too aggressively. Once we believe that an arm is optimal with reasonably high confidence, computation will be heavily focused on evaluating this arm even though we need to become more confident about the sub-optimality of competing models to improve our confidence level. This criticism motivates our chosen algorithm TTTS (summarised in Algorithm 2), where instead of sampling a single model according to π, we sample two distinct models. We then uniformly choose between these two models for the next evaluation, allowing a greater exploration of the arms and much improved rates of convergence to the desired confidence level (Russo, 2016). We use this new evaluation to update our belief and continue making evaluations until we believe that a model is optimal with a higher probability than $1-\delta$ and terminate the algorithm. An example run of TTTS is demonstrated on a synthetic example in Figure 2, where we simulate from 5 Gaussian distributions with means $\{0.65, 0.69, 0.69, 0.70, 0.71\}$ and standard deviation 0.01 to mimic accuracy measurements for a model selection problem.

We now explain how we calculate π (our be-

[3] We enforce a minimum of three evaluations to ensure that the t distribution in our posterior remains well-defined

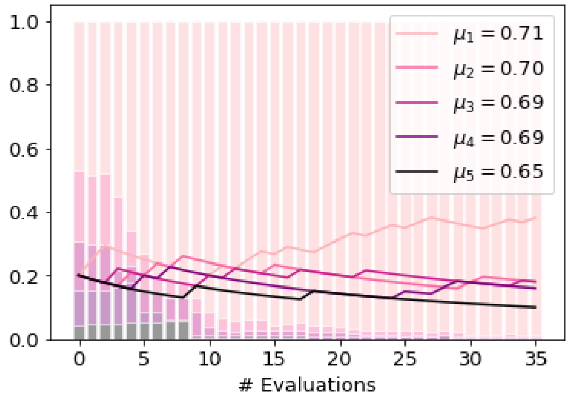

Figure 2: TTTS seeking the optimal model with confidence 0.99 from 5 synthetic models. The background represents our evolving belief π in the optimal model and the lines represent the proportion of the total evaluations made on each model. We start evaluating the models uniformly but our adaptive algorithm quickly focuses resources on the best models.

lief in the location of the optimal model) using well-known results from Bayesian decision theory (see Berger (2013) for a comprehensive coverage). As justified earlier, we assume that the evaluations of model m are independently distributed with a Gaussian distribution $\mathcal{N}(\mu_m, \sigma_m^2)$ for some unknown mean μ_m and variance σ_m^2. Although we are primarily interested in learning μ_m, we must also learn σ_m^2 in order to make confidence guarantees about the optimality of our selected model. Therefore, as well as keeping track of the sample means for the evaluations of each model $\hat{\mu}_m$, we also keep track of the sample variances $\hat{S}_m$ and counters T_m of the number of times each model has been evaluated. To facilitate inference, we choose a uniform prior for the unknown μ_m and σ_m. Not only is this a conjugate prior for Gaussian likelihoods, but it is also shown to encourage beneficial exploratory behaviour when using Thompson sampling on Gaussian bandit problems (Honda and Takemura, 2014) and so allows fast identification of optimal arms (or models). After observing T_m evaluations of each model and producing estimates $\hat{\mu}_m$ and $\hat{S}_m$, our posterior belief for each deviation between the true and observed model means $\mu_m - \hat{\mu}_m$ satisfies (as derived in (Honda and Takemura, 2014));

$$\sqrt{\frac{T_m(T_m-2)}{\hat{S}_m}}\,(\mu_m - \hat{\mu}_m)\,|\,\hat{\mu}_m, \hat{S}_m \sim t_{T_m-2},$$

where t_d is a Student's t-distribution with d degrees of freedom.

π is then defined as the probability vector, such that π_m is the relative probability that μ_m is the largest according to this posterior belief. Unfortunately, there is no closed form expression for the maximum of N t-distributions and so FIESTA uses a simple Monte-Carlo approximation based on the sample maxima of repeated draws from our posteriors for μ_m. In practice this is very accurate and did not slow down our experiments, especially in comparison to the time saved by reducing the number of model evaluations.

4.3 Batch Fixed Confidence by BTS

NLP practitioners often have the computational capacity to fit models in parallel across multiple workers, evaluating multiple models or the same model across multiple seeds at once. Their model selection routines must therefore provide batches of models to evaluate. Our proposed solution to FB model selection naturally provides such batches, with each successive round of SH producing a collection of model evaluations that can be calculated in parallel. Unfortunately, TTTS for FC model selection successively chooses and then waits for the evaluation of single models and so is not naturally suited to parallelism.

Extending TTTS to batch decision making is an open problem in the MAB literature. Therefore, we instead consider **batch Thompson sampling** (BTS), an extension of standard Thompson sampling (as described in Section 4.2) to batch sampling from the related field of Bayesian optimisation (Kandasamy et al., 2018). At each step in our selection process we take B model draws according to our current belief π that the model is optimal, where B represents our computational capacity. This is in contrast to the single draw in standard Thompson sampling and the drawn pair in TTTS. In addition, this approach extends to the asynchronous setting, where rather than waiting for the whole batch of B models to be evaluated before choosing the next batch, each worker can draw a new model to evaluate according to the updated π. This flexibility provides an additional efficiency gain for problems where the different models have a wide range of run times.

5 Experiments

We now test our three algorithms on a challenging model selection task typical of NLP, selecting between eight Target Dependent Sentiment Analysis (TDSA) models based on their macro F1 score. We consider two variants of four re-implementations of well-known TDSA models: ATAE (Wang et al., 2016), IAN (Ma et al., 2017), TDLSTM (Tang et al., 2016) (without target words in the left and right LSTM), and a non-target-aware LSTM method used as the baseline in Tang et al. (2016).

These methods represent *state-of-the-art* within TDSA, with only small differences in performance between TDLSTM, IAN, and ATAE (see figure 3). All the models are re-implemented in PyTorch (Paszke et al., 2017) using AllenNLP (Gardner et al., 2018). To ensure the only difference between the models is their network architecture the models use the same optimiser settings and the same regularisation. All words are lower cased and we use the same Glove common crawl 840B token 300 dimension word embedding (Pennington et al., 2014). We use variational (Gal and Ghahramani, 2016) and regular (Hinton et al., 2012) dropout for regularisation and an ADAM (Kingma and Ba, 2014) optimiser with standard settings, a batch size of 32 and use at most 100 epochs (with early stopping on a validation set). Many of these settings are not the same as originally implemented, however, having the same training setup is required for fair comparison (this explains the differences between our results and the original implementations). To increase the difficulty of our model selection problem, we additionally create four extra models by reducing the dimensions of the Glove vectors to 50 and removing dropout. Although these models are clearly not *state-of-the-art*, they increase the size of our candidate model set and so provide a more complicated model selection problem (an intuition discussed in Appendix A).

All of the TDSA experiments are conducted on the well-studied SemEval 2014 task 4 Restaurant dataset (Pontiki et al., 2014) and we force train-val-test splits to follow the same ratios as this dataset's official train-test split. Each individual model evaluation is then made on a randomly generated train-test split and random seed to access both sources of evaluation variability.

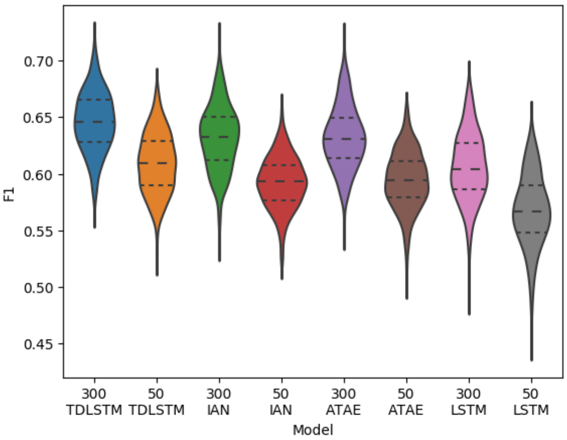

Figure 3: F1 scores for our candidate TDSA models. After 500 evaluations of each model on different data splits and model seeds we see that the TDLSTM is the *state-of-the-art* model.

5.1 Fixed Budget Model Selection

We use the TDSA model selection problem to test fixed budget model selection. To thoroughly test our algorithm, we consider an additional four models based on 200 dimensional Glove vectors, bringing the total number of models to 12. We compare our approach of sequential halving to the standard non-adaptive approach of splitting the available computational budget equally between the 12 candidate models. For example, we would allocate a budget of 24 model evaluations as evaluating each model two times and selecting the model with the highest sample mean.

Figure 4 compares the proportion of $10,000$ runs of sequential halving that correctly identify the optimal model with the proportion identified by the non-adaptive approach with the same computational budget. Sequential halving identifies the optimal model more reliably ($\approx$ 15% more often) than the current approach to FB model selection in NLP. Using sequential halving with 204 evaluations almost always (99% of runs) selects the optimal model, whereas the non-adaptive approach is only correct 85% of the time.

5.2 Fixed Confidence Model Selection

We perform fixed confidence model selection on the eight TDSA candidate models (the full models and those based on 50 dimensional vectors). We compare TTTS to a non-adaptive approach where all models are evaluated at each step, irrespective of the results of earlier evaluations (the standard

	# evaluations with Non-Adaptive				# evaluations with TTTS			
δ	min	mean	max	% correctly selected	min	mean	max	% correctly selected
0.05	48	281	1552	100	27	130	518	100
0.1	40	206	1192	99	24	96	460	99
0.2	32	128	608	96	24	65	274	97

Table 2: Number of evaluations required to select a TDSA model at a range of confidence levels across 500 runs of TTTS and a standard non-adaptive approach.

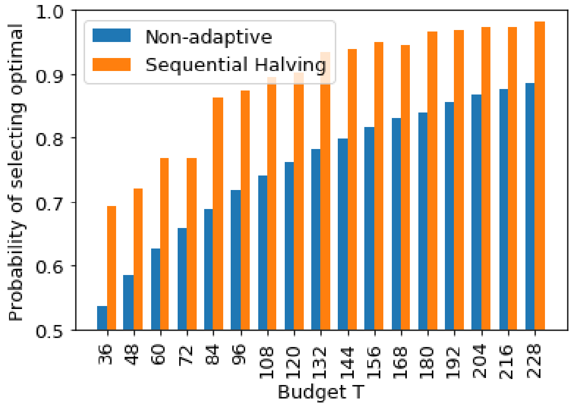

Figure 4: Proportion of the runs correctly selecting the optimal TDSA model using sequential halving against the standard non-adaptive approach. Sequential halving consistently identifies the optimal model at a significantly higher rate across a wide range of budgets.

approach for model selection in NLP). We run this non-adaptive approach until we reach the required confidence level calculated using the same Bayesian framework as in TTTS.

We run each approach 500 times and note the number evaluations required to get to a range of confidence levels (Table 2) alongside the proportion that correctly identify the optimal model. TTTS requires substantially less model evaluations (in terms of the minimum, mean and max across our runs) to reach a given confidence level than the non-adaptive approach, achieving the same reliability at half the cost (on average). TTTS is able to quickly identify sub-optimal models and so can avoid wasting resources repeatedly evaluating the whole candidate set.

Finally, we test our proposed approach to batch FC model selection by running exactly the same experiment but using BTS to choose collections of four and eight models at a time (Table 3). As expected, performance degrades as we increase batch size, with batches of four allowing more fine grained control over model evaluations than using batches of eight. In particular, due to the exploitative nature of Thompson sampling, we see that selecting models to a very high confidence (95%) requires more computation with BTS than the standard non-adaptive approach. However, BTS does reach the other confidence levels faster and correctly identifies the optimal model more often. However, as TTTS performs significantly better across all confidence levels, we emphasise the need for a less-exploitative version of BTS with adjustments similar to those used in TTTS.

6 Conclusions

The aim of this paper has been to propose three algorithms for model selection in NLP, providing efficient and reliable selection for two distinct realistic scenarios: fixed confidence and fixed budget model selection. Crucially, our research further calls into question the current practice in NLP evaluation as used in the literature and international competitions such as SemEval. Our algorithms adaptively allocate resources to evaluate promising models, basing evaluations across multiple random seeds and train-test splits. We demonstrate that this allows significant computational savings and improves reliability over current model selection approaches.

Although we have demonstrated that our algorithms perform well on a complex model selection problem typical of NLP, there is still work to be done to create algorithms more suited to these problems. Future research directions include making selection routines more robust to evaluation outliers, relaxing our Gaussian assumptions and developing more effective batch strategies.

7 Acknowledgements

The authors are grateful to reviewers, whose comments and advice have greatly improved this paper. The research was supported by an EPSRC

	#	evaluations	with	BTS-4	#	evaluations	with	BTS-8
δ	min	mean	max	% correctly selected	min	mean	max	% correctly selected
0.05	28	282	1392	100	88	315	1128	100
0.1	24	144	520	100	56	178	784	100
0.2	24	76	280	98	32	106	352	99

Table 3: Number of evaluations of required to select a TDSA model at a range of confidence levels across 500 runs of BTS selecting batches of 4 and 8 models.

Doctoral Training Grant and the STOR-i Centre for Doctoral Training. We thank Dr Chris Jewell at the Centre for Health Informatics, Computing, and Statistics, Lancaster University for the loan of a NVIDIA GP100-equipped workstation for this study.

References

Andrea Arcuri and Lionel Briand. 2014. A hitchhiker's guide to statistical tests for assessing randomized algorithms in software engineering. *Software Testing, Verification and Reliability*, 24(3):219–250.

Jean-Yves Audibert and Sébastien Bubeck. 2010. Best arm identification in multi-armed bandits. In *COLT - 23th Conference on Learning Theory - 2010*, pages 13–p.

James O Berger. 2013. *Statistical decision theory and Bayesian analysis*. Springer Science & Business Media.

Rotem Dror, Gili Baumer, Segev Shlomov, and Roi Reichart. 2018. The hitchhiker's guide to testing statistical significance in natural language processing. In *Proceedings of the 56th Annual Meeting of the Association for Computational Linguistics (Volume 1: Long Papers)*, pages 1383–1392. Association for Computational Linguistics.

Bradley Efron and Robert J Tibshirani. 1994. *An introduction to the bootstrap*. CRC press.

Eyal Even-Dar, Shie Mannor, and Yishay Mansour. 2002. Pac bounds for multi-armed bandit and markov decision processes. In *International Conference on Computational Learning Theory*, pages 255–270. Springer.

Jerome Friedman, Trevor Hastie, and Robert Tibshirani. 2001. *The elements of statistical learning*. Springer series in statistics New York, NY, USA:.

Yarin Gal and Zoubin Ghahramani. 2016. A theoretically grounded application of dropout in recurrent neural networks. In D. D. Lee, M. Sugiyama, U. V. Luxburg, I. Guyon, and R. Garnett, editors, *Advances in Neural Information Processing Systems 29*, pages 1019–1027. Curran Associates, Inc.

Matt Gardner, Joel Grus, Mark Neumann, Oyvind Tafjord, Pradeep Dasigi, Nelson F. Liu, Matthew Peters, Michael Schmitz, and Luke Zettlemoyer. 2018. Allennlp: A deep semantic natural language processing platform. In *Proceedings of Workshop for NLP Open Source Software (NLP-OSS)*, pages 1–6. Association for Computational Linguistics.

Aurélien Garivier and Emilie Kaufmann. 2016. Optimal best arm identification with fixed confidence. In *Conference on Learning Theory*, pages 998–1027.

Gholamreza Haffari, Tuan Dung Tran, and Mark Carman. 2017. Efficient benchmarking of nlp apis using multi-armed bandits. In *Proceedings of the 15th Conference of the European Chapter of the Association for Computational Linguistics: Volume 1, Long Papers*, pages 408–416. Association for Computational Linguistics.

Geoffrey E Hinton, Nitish Srivastava, Alex Krizhevsky, Ilya Sutskever, and Ruslan R Salakhutdinov. 2012. Improving neural networks by preventing co-adaptation of feature detectors. *arXiv preprint arXiv:1207.0580*.

Junya Honda and Akimichi Takemura. 2014. Optimality of thompson sampling for gaussian bandits depends on priors. In *Artificial Intelligence and Statistics*, pages 375–383.

Kevin Jamieson, Matthew Malloy, Robert Nowak, and Sebastien Bubeck. 2013. On finding the largest mean among many. *arXiv preprint arXiv:1306.3917*.

Kevin Jamieson, Matthew Malloy, Robert Nowak, and Sébastien Bubeck. 2014. lilucb: An optimal exploration algorithm for multi-armed bandits. In *Conference on Learning Theory*, pages 423–439.

Kevin Jamieson and Robert Nowak. 2014. Best-arm identification algorithms for multi-armed bandits in the fixed confidence setting. In *2014 48th Annual Conference on Information Sciences and Systems (CISS)*, pages 1–6. IEEE.

Kirthevasan Kandasamy, Akshay Krishnamurthy, Jeff Schneider, and Barnabás Póczos. 2018. Parallelised bayesian optimisation via thompson sampling. In *International Conference on Artificial Intelligence and Statistics*.

Zohar Karnin, Tomer Koren, and Oren Somekh. 2013. Almost optimal exploration in multi-armed bandits. In *International Conference on Machine Learning*, pages 1238–1246.

Diederik P Kingma and Jimmy Ba. 2014. Adam: A method for stochastic optimization. *arXiv preprint arXiv:1412.6980*.

Ron Kohavi. 1995. A study of cross-validation and bootstrap for accuracy estimation and model selection. In *Proceedings of the 14th international joint conference on Artificial intelligence-Volume 2*, pages 1137–1143. Morgan Kaufmann Publishers Inc.

Tze Leung Lai and Herbert Robbins. 1985. Asymptotically efficient adaptive allocation rules. *Advances in applied mathematics*, 6(1):4–22.

Guillaume Lample, Miguel Ballesteros, Sandeep Subramanian, Kazuya Kawakami, and Chris Dyer. 2016. Neural architectures for named entity recognition. In *Proceedings of the 2016 Conference of the North American Chapter of the Association for Computational Linguistics: Human Language Technologies*, pages 260–270. Association for Computational Linguistics.

Carolin Lawrence, Artem Sokolov, and Stefan Riezler. 2017. Counterfactual learning from bandit feedback under deterministic logging : A case study in statistical machine translation. In *Proceedings of the 2017 Conference on Empirical Methods in Natural Language Processing*, pages 2566–2576. Association for Computational Linguistics.

Lihong Li, Wei Chu, John Langford, and Robert E Schapire. 2010. A contextual-bandit approach to personalized news article recommendation. In *Proceedings of the 19th international conference on World wide web*, pages 661–670. ACM.

Dehong Ma, Sujian Li, Xiaodong Zhang, and Houfeng Wang. 2017. Interactive attention networks for aspect-level sentiment classification. In *Proceedings of the 26th International Joint Conference on Artificial Intelligence*, pages 4068–4074. AAAI Press.

Xuezhe Ma and Eduard Hovy. 2016. End-to-end sequence labeling via bi-directional lstm-cnns-crf. In *Proceedings of the 54th Annual Meeting of the Association for Computational Linguistics (Volume 1: Long Papers)*, pages 1064–1074. Association for Computational Linguistics.

Shie Mannor and John N Tsitsiklis. 2004. The sample complexity of exploration in the multi-armed bandit problem. *Journal of Machine Learning Research*, 5(Jun):623–648.

Tomas Mikolov, Ilya Sutskever, Kai Chen, Greg S Corrado, and Jeff Dean. 2013. Distributed representations of words and phrases and their compositionality. In *Advances in neural information processing systems*, pages 3111–3119.

Henry Moss, David Leslie, and Paul Rayson. 2018. Using j-k-fold cross validation to reduce variance when tuning nlp models. In *Proceedings of the 27th International Conference on Computational Linguistics*, pages 2978–2989. Association for Computational Linguistics.

Khanh Nguyen, Hal Daumé III, and Jordan Boyd-Graber. 2017. Reinforcement learning for bandit neural machine translation with simulated human feedback. In *Proceedings of the 2017 Conference on Empirical Methods in Natural Language Processing*, pages 1464–1474. Association for Computational Linguistics.

Adam Paszke, Sam Gross, Soumith Chintala, Gregory Chanan, Edward Yang, Zachary DeVito, Zeming Lin, Alban Desmaison, Luca Antiga, and Adam Lerer. 2017. Automatic differentiation in pytorch. In *NIPS-W*.

Jeffrey Pennington, Richard Socher, and Christopher Manning. 2014. Glove: Global vectors for word representation. In *Proceedings of the 2014 Conference on Empirical Methods in Natural Language Processing (EMNLP)*, pages 1532–1543. Association for Computational Linguistics.

Maria Pontiki, Dimitris Galanis, John Pavlopoulos, Harris Papageorgiou, Ion Androutsopoulos, and Suresh Manandhar. 2014. Semeval-2014 task 4: Aspect based sentiment analysis. In *Proceedings of the 8th International Workshop on Semantic Evaluation (SemEval 2014)*, pages 27–35. Association for Computational Linguistics.

Nils Reimers and Iryna Gurevych. 2017. Reporting score distributions makes a difference: Performance study of lstm-networks for sequence tagging. In *Proceedings of the 2017 Conference on Empirical Methods in Natural Language Processing*, pages 338–348. Association for Computational Linguistics.

Nils Reimers and Iryna Gurevych. 2018. Why comparing single performance scores does not allow to draw conclusions about machine learning approaches. *arXiv preprint arXiv:1803.09578*.

Daniel Russo. 2016. Simple bayesian algorithms for best arm identification. In *Conference on Learning Theory*, pages 1417–1418.

Daniel J Russo, Benjamin Van Roy, Abbas Kazerouni, Ian Osband, Zheng Wen, et al. 2018. A tutorial on thompson sampling. *Foundations and Trends in Machine Learning*, 11(1):1–96.

SemEval. 2018. *Proceedings of The 12th International Workshop on Semantic Evaluation*. Association for Computational Linguistics, New Orleans, Louisiana.

Artem Sokolov, Julia Kreutzer, Christopher Lo, and Stefan Riezler. 2016. Learning structured predictors from bandit feedback for interactive nlp. In *Proceedings of the 54th Annual Meeting of the Association for Computational Linguistics (Volume 1: Long Papers)*, pages 1610–1620. Association for Computational Linguistics.

Duyu Tang, Bing Qin, Xiaocheng Feng, and Ting Liu. 2016. Effective lstms for target-dependent sentiment classification. In *Proceedings of COLING 2016, the 26th International Conference on Computational Linguistics: Technical Papers*, pages 3298–3307. The COLING 2016 Organizing Committee.

Sofía S Villar, Jack Bowden, and James Wason. 2015. Multi-armed bandit models for the optimal design of clinical trials: benefits and challenges. *Statistical science: a review journal of the Institute of Mathematical Statistics*, 30(2):199.

Yequan Wang, Minlie Huang, xiaoyan zhu, and Li Zhao. 2016. Attention-based lstm for aspect-level sentiment classification. In *Proceedings of the 2016 Conference on Empirical Methods in Natural Language Processing*, pages 606–615. Association for Computational Linguistics.

Jie Yang, Shuailong Liang, and Yue Zhang. 2018. Design challenges and misconceptions in neural sequence labeling. In *Proceedings of the 27th International Conference on Computational Linguistics*, pages 3879–3889. Association for Computational Linguistics.

8 Appendix

A Characterising the Difficulty of a Model Selection Problem

We briefly summarise a result from the best-arm identification literature, providing intuition for our experiment section by providing a mechanism to characterise the difficulty of a model selection problem. Intuitively, model selection difficulty increases with the size of the set of candidate models N and as the performance of sub-optimal models approaches that of the optimal model (and becomes harder to distinguish), i.e. as $\mu_{m^*} - \mu_m$ gets small for some sub-optimal arm m. In fact, it is well known in the MAB literature that it is exactly these two properties that characterise the complexity of a best-arm-identification problem, confirming our intuition for model selection. Mannor and Tsitsiklis (2004) show that the number of arm pulls required for the identification of a best arm at a confidence level $1 - \delta$ has at least a computational complexity of $O(H \log(1/\delta))$, where

$$H = \sum_{m' \in S \backslash \{m*\}} \frac{1}{\left(\mu_{m^*} - \mu_m\right)^2}.$$