Efficient Benchmarking of NLP APIs using Multi-armed Bandits

Gholamreza Haffari and **Tuan Tran** and **Mark Carman**
Faculty of Information Technology, Monash University
Clayton, VICTORIA 3800, Australia
{gholamreza.haffari, mark.carman}@monash.edu
tdtra18@student.monash.edu

Abstract

Comparing NLP systems to select the best one for a task of interest, such as named entity recognition, is critical for practitioners and researchers. A rigorous approach involves setting up a hypothesis testing scenario using the performance of the systems on query documents. However, often the hypothesis testing approach needs to send a large number of document queries to the systems, which can be problematic. In this paper, we present an effective alternative based on the multi-armed bandit (MAB). We propose a hierarchical generative model to represent the uncertainty in the performance measures of the competing systems, to be used by Thompson Sampling to solve the resulting MAB. Experimental results on both synthetic and real data show that our approach requires significantly fewer queries compared to the standard benchmarking technique to identify the best system according to F-measure.

1 Introduction

F-measureAs new NLP systems are (continually) introduced for a task of interest, such as named entity recognition (NER), it is crucial for practioneers and researchers to select the best system. These systems may be designed based on different models and/or learning algorithms. For instance, due to recent advancement in NER research, several NER systems have been proposed and then supported in APIs such as OpenNLP (Ingersoll et al., 2013), Stanford NER (Finkel et al., 2005), ANNIE (Cunningham et al., 2002) and Meaning Cloud (MeaningCloud-LLC, 1998) to name a few.

Often, the competing NLP systems are benchmarked according to their average performance measure, e.g. F-measure capturing both Precision and Recall, across a set of example documents. Each document produces a single F-measure and the true performance of the system is considered to be the expected value across all possible documents from the domain. Performance on individual documents correspond to samples from the performance distribution of the system, and can then be used to determine the best system (or set of systems should the highest performing system not be unique) using rigorous hypothesis testing. However, this approach usually requires querying each competing system with a large number of documents, which can be problematic if either the number of test documents is limited, or the systems are implemented as APIs by a third party and performing each query incurs a cost.

In this paper, we present a statistically effective method to identify the best system from a pool of systems. Our approach requires significantly fewer example documents to reach similar guarantees as the traditional hypothesis testing set up, hence reducing the cost and increasing the speed of inference. Inspired by the previous work (Scott, 2015; Gabillon et al., 2012; Maron and Moore, 1993), We formulate the benchmarking problem as a sequential decision process of choosing the best arm as the results of new queried documents are received. More specifically, our formulation is based on the best arm identification in a multi-armed bandit (MAB) decision process. This allows us to adapt Thompson Sampling (Thompson, 1933) and its variants (Russo, 2016) to efficiently solve the resulting MAB problem.

A crucial difference between the MAB approach and the traditional hypothesis testing is that it is a *sequential* testing process, instead of a static testing process which forces the benchmarker to wait for a *final answer* at the end of an experiment. As such, we need to model the uncertainty regarding the estimated F-measure of each competing system, and continually update it as each new document is queried. We propose a novel hierarchical model for this purpose, which is generally applicable to document-level evaluation tasks based on F-measure. The inference in our model is done using standard sampling techniques, such as Gibbs sampling.

We analyse empirically the performance of our approach versus the standard hypothesis testing baselines on synthetic datasets as well as real data for the tasks of sentiment classification and named entity recognition. The empirical results confirm that the number of query documents needed to achieve a particular statistical significance level with our approach is much lower than that required by the hypothesis testing baselines.

2 Best Arm Selection in Multi-Armed Bandit

Our aim is to identify the best system among a finite set of systems based on the noisy sequential measurements of their quality. We formulate this problem as the best arm selection in multi-armed bandit. MAB is a sequential decision process where at each time step n an arm a_n from the collection of K slot machines is chosen and played by the gambler. Each arm $a \in \{1, \ldots, K\}$ is associated with an *unknown* reward distribution $f(y|\boldsymbol{\theta}_a)$ from which the reward is generated when the arm is pulled. In the best arm selection problem, the gambler's goal is to select the arm which has the highest expected reward.

In the common formulation of the MAB, the gambler wants to maximise his cumulative rewards. Intuitively, maximising cumulative rewards eventually leads to the selection of the best arm since it is the optimal decision. However, (Bubeck et al., 2009) gives a theoretical analysis that any strategies for optimising cumulative reward is suboptimal in identifying the best performing arm. To this end, several algorithms have been proposed for the best arm selection e.g. (Maron and Moore, 1993; Gabillon et al., 2012; Russo, 2016). Although originally developed for maximising cumulative rewards, (Chapelle and Li, 2011; Scott, 2015) provide extensive empirical evidence for the practical success of the Thompson Sampling algorithm for the best arm selection. In what follows, we present Thompson Sampling (TS) and one of its variants, called Pure Exploration TS (PETS), designed specifically for the best arm selection.

2.1 Thompson Sampling

Let us denote by $(a_1, y_1), \ldots, (a_T, y_T)$ the sequence of pulled arms and the revealed rewards, a_t is the arm pulled at time step t and y_t is its associated reward. Note that this sequence is continually growing as the experiment progresses and new arms are pulled. Let $f(y|\boldsymbol{\theta}_a)$ be the probability distribution to model the unknown reward function of the arm a. Had we known the parameters of the reward functions, the best arm could then be selected as

$$\operatorname{argmax}_a E_{f(y|\boldsymbol{\theta}_a)}[y] = \operatorname{argmax}_a \int f(y|\boldsymbol{\theta}_a) y dy.$$

Let us denote the collection of all parameters by $\Theta := (\boldsymbol{\theta}_1, \ldots, \boldsymbol{\theta}_K)$. Assuming a prior over the parameters $\pi_0(\Theta)$, we take a Bayesian approach and reason about the posterior of the parameters:

$$\pi_T(\Theta) = \frac{\pi_0(\Theta) L_T(\Theta)}{\int_{\boldsymbol{\Theta}'} \pi_0(\Theta') L_T(\Theta') d\Theta'}$$

where $\boldsymbol{\Theta}$ is the parameter domain, and $L_T(\Theta)$ is the likelihood of the observed data $\{(a_t, y_t)\}_1^T$

$$L_T(\Theta) := \prod_{t=1}^{T} f(y_t|\boldsymbol{\theta}_{a_t}).$$

The posterior probability that a particular arm a is optimal (i.e. has the highest expected reward) is:

$$\alpha_{T,a} := \int_{\boldsymbol{\Theta}_a} \pi_T(\Theta) d\Theta$$

where $\boldsymbol{\Theta}_a$ is the set of those parameter values under which the arm a would be selected as the optimal arm:

$$\boldsymbol{\Theta}_a := \{\Theta \in \boldsymbol{\Theta} | E_{f(y|\boldsymbol{\theta}_a)} = \operatorname{argmax}_{a'} E_{f(y|\boldsymbol{\theta}_{a'})}\}$$

In Thompson Sampling the next arm to pull is sampled according to the posterior probability of the arm being optimal. That is, an arm a is selected with probability $\alpha_{T,a}$. Efficient implementation of Thompson Sampling generates a sample from $\alpha_{T,a}$ indirectly by first generating a sample $\hat{\Theta}$ from $\pi_n(\Theta)$ and then selecting the next arm to pull by $\operatorname{argmax}_a E_{f(y|\hat{\boldsymbol{\theta}}_a)}[y]$.

Algorithm 1 Pure Exploration TS (PETS)

```
Initialization: Pull each arm once
$t = K$
while termination condition is not met do
    $\hat{\Theta} \sim \pi_t(\Theta)$
    $a \leftarrow \text{argmax}_k \, E_{f(y|\hat{\theta}_k)}[y]$
    $r \sim \text{uniform}(0, 1)$
    if $r \leq \beta$ then
        Pull $a$ and update the posterior $\pi_{t+1}(\Theta)$
    else
        $b \leftarrow a$
        while $b = a$ do
            $\hat{\Theta} \sim \pi_t(\Theta)$
            $b \leftarrow \text{argmax}_k \, E_{f(y|\hat{\theta}_k)}[y]$
        end while
        Pull $b$ and update the posterior $\pi_{t+1}(\Theta)$
    end if
    $t \leftarrow t + 1$
end while
```

2.2 Pure Exploration Thompson Sampling

Thompson sampling can perform poorly for the best arm identification problem. The reason is that once it discovers a particular arm is performing well, it becomes overconfident and almost always selects that arm in the future iterations. In case that arm is not the best arm, it takes a long time for the algorithm to divert from it. For example, if $\alpha_{T,a} = 90\%$, then the algorithm selects an arm other than a on average on every 10 iterations, which would make it significantly longer to get to a point where $\alpha_{T',a'} = 95\%$, i.e. the point where the algorithm terminates with confidence 95% in a different arm a'.

Let $\boldsymbol{\alpha}_T := (\alpha_{T,1}, \ldots, \alpha_{T,K})$ be the vector of arm probabilities to be optimal. Pure Exploration Thompson Sampling (Russo, 2016) addresses the above deficiency of Thompson Sampling by throwing away, with probability β, the arm a sampled from $\boldsymbol{\alpha}_T$. Instead, it samples another arm $b \neq a$ with the probability proportional to $\alpha_{T,b}$. The exploration parameter β prevents the algorithm from exclusively focusing on one arm. Usually β is set to .5 but we empirically investigate other values for this parameter in §4. We can revert to basic Thompson Sampling by setting $\beta = 1$ in PETS. Similar to Thompson Sampling, this arm selection method can be efficiently implemented as shown in Algorithm 1. We terminate the algorithm when it reaches a maximum number of queries or when $\alpha_{T,a} \geq 1 - \delta$, where δ is the confidence parameter provided in the input.

3 A Probabilistic Generative Model of F-measure

In this section we present a novel hierarchical Bayesian model for capturing the uncertainty over systems' F-measures, as the prediction outcome on new query documents are received. We present this model for F-measure, however, we note that it can be extended for other performance measures as well.

F-measure is defined as the harmonic mean of the precision and recall:

$$\begin{aligned} \text{F-measure} &:= \frac{2}{\frac{1}{\text{precision}} + \frac{1}{\text{recall}}} \\ \text{precision} &:= \frac{TP}{TP + FP} \;, \; \text{recall} := \frac{TP}{TP + FN} \end{aligned}$$

where (TP, FP, TN, FN) denote true positive, false positive, true negative, and false negative counts. These counts result from comparing the predictions of a system with the ground truth annotations, and they sum to the total number of annotated data items N. We denote the normalised version of the counts by *rates* $(\hat{TP}, \hat{FP}, \hat{TN}, \hat{FN})$, which are derived by dividing the raw counts by N. Importantly, the rate statistics are enough to calculate precision, recall, and F-measure.

Instead of modelling the uncertainty over the F-measure of a system directly, we model the uncertainty over its rate statistics. Any distribution over $(\hat{TP}, \hat{FP}, \hat{TN}, \hat{FN})$ then induces a distribution over F-measure. The benefit of working with the rate statistics is that they relate more naturally to the observed (TP, FP, TN, FN) counts, as established in our generative model in the following.

More specifically, we assume a hierarchical model to generate the rate statistics of the systems and the observed (TP_d, FP_d, TN_d, FN_d) counts over a collection of documents $d \in \mathcal{D}$. For each system, we draw its rate statistics $\boldsymbol{\theta}_k := (\hat{TP}_k, \hat{FP}_k, \hat{TN}_k, \hat{FN}_k)$ from a Dirichlet prior. To generate the counts statistics resulting from applying the system a_t on the document d_t, we first generate a document-specific rate vector $\boldsymbol{\mu}_{d_t}$ from a Dirichlet distribution centred around $\boldsymbol{\theta}_{a_t}$. Note that including explicit document-specific rates $\boldsymbol{\mu}_{d_t}$ in the model (from which the binomial counts are drawn) is necessary in order to allow for sufficient variation in the observed error rates across documents, due to the inherent differences in difficulty of labelling different documents.[1]

[1]In other words, in order to model the observed vari-

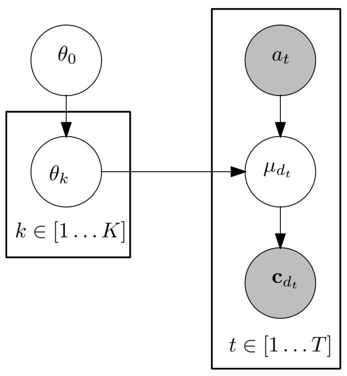

Figure 1: The graphical model for the probabilistic generation of a system's parameters $\boldsymbol{\theta_k}$ and a document's counts $\mathbf{c}_{d_t}$, as the selected system a_t is applied onto the document d_t at the time step t. The observed quantities are shaded.

Algorithm 2 Identifying the best system

```
Require: $K$: Number of arms, $J$: Number of samples from posterior $\pi(.)$, $\mathcal{D}$: Document collection,
    Fmeasure$(\hat{TP}, \hat{TN}, \hat{FP}, \hat{FN}) := \frac{2\hat{TP}}{2\hat{TP}+\hat{FP}+\hat{FN}}$,
    NextDoc$(a, \mathcal{D})$: Next document for an arm $a$ from $\mathcal{D}$
 1: for $k \in [1..K]$ do
 2:     $\mathcal{D}_k \leftarrow$ NextDoc$(k, \mathcal{D})$
 3:     $\mathcal{S}_k \leftarrow \{\tilde{\boldsymbol{\theta}}_j | \forall j \in [1..J] : \tilde{\boldsymbol{\theta}}_j \overset{\text{Gibbs}}{\sim} \pi(\boldsymbol{\theta}_j | \mathcal{D}_k)\}$
 4: end for
 5: while termination condition is not met do
 6:     for $k \in [1..K]$ do
 7:         $f_k \sim \{$Fmeasure$(\tilde{\boldsymbol{\theta}}) | \tilde{\boldsymbol{\theta}} \in \mathcal{S}_k\}$
 8:     end for
 9:     $a \leftarrow \text{argmax}_k f_k$
10:     $r \sim$ uniform$(0, 1)$
11:     $b \leftarrow a$
12:     if $r > \beta$ then
13:         while $b = a$ do
14:             for $k \in [1..K]$ do
15:                 $f_k \sim \{$Fmeasure$(\tilde{\boldsymbol{\theta}}) | \tilde{\boldsymbol{\theta}} \in \mathcal{S}_k\}$
16:             end for
17:             $b \leftarrow \text{argmax}_k f_k$
18:         end while
19:     end if
20:     $\mathcal{D}_b \leftarrow \mathcal{D}_b \cup$ NextDoc$(b, \mathcal{D})$
21:     $\mathcal{S}_b \leftarrow \{\tilde{\boldsymbol{\theta}}_j | \forall j \in [1..J] : \tilde{\boldsymbol{\theta}}_j \overset{\text{Gibbs}}{\sim} \pi(\boldsymbol{\theta}_j | \mathcal{D}_b)\}$
22: end while
```

We then generate the observed counts $\mathbf{c}_{d_t} := (TP_{d_t}, FP_{d_t}, TN_{d_t}, FN_{d_t})$ from the Bionomial distribution with parameters $\boldsymbol{\mu}_{d_t}$ and N_{d_t}, where N_{d_t} is the number data items in d_t. In summary, the generative model is as follows:

$$\begin{aligned} \forall k \in [1..K] \quad &: \quad \boldsymbol{\theta}_k \sim \text{Dirichlet}(\boldsymbol{\theta}_0, \alpha_0) \\ \forall t \in [1..T] \quad &: \quad \boldsymbol{\mu}_{d_t} \sim \text{Dirichlet}(\boldsymbol{\theta}_{a_t}, \alpha) \\ & \qquad \mathbf{c}_{d_t} \sim \text{Bionomial}(\boldsymbol{\mu}_{d_t}, N_{d_t}) \end{aligned}$$

where α_0 and α are the concentration parameters, which we set to 1 in our experiments in §4. Figure 1 depicts the graphical model.

For inference, the quantities of interest are the unknown rates for the systems $\{\boldsymbol{\theta}_k\}_{k=1}^{K}$. The observed quantities are document-specific counts $\{\mathbf{c}_{d_t}\}_{t=1}^{T}$, and we would like to marginalise out the latent document-specific rate variables $\{\boldsymbol{\mu}_{d_t}\}_{t=1}^{T}$. We resort to Gibbs sampling for inference in our model. That is, we iteratively select a hidden variable and sample a value from its posterior given all the other variables are fixed to their current values. In our experiments, we collect 1000 samples from the posterior.[2] Algorithm 2 depicts the sampling-based inference for the posterior embedded in the PETS algorithm for the best system identification.

F-measure is a frequently used evaluation measure, which can straightforwardly be parametrized to allows for varying the importance of precision versus recall:

$$\text{F}_\beta\text{-measure} := \frac{2}{\frac{\beta}{\text{precision}} + \frac{1-\beta}{\text{recall}}}$$

where β is a parameter trading off precision and recall. We note that our approach can be applied straightforwardly to F$_\beta$-measure to put more weight on precision or recall where appropriate.

4 Empirical Results and Analysis

We designed two sets of experiments to examine the efficiency and performance of each algorithm using synthetic data as well as real data for sentence level sentiment classification and named entity recognition tasks. With the synthetic data, we analyse our probabilistic generative model for F-measure in combination with the arm selection algorithms. With the real data, we showcase the statistical efficiency of our best system identification approach compared to the standard hypothesis testing approach (Demšar, 2006).

In the real data experiments, we define the "best system" as the system with the highest F-measure based on all documents in the collection. The "success rate" in the NER/Sentiment tasks is then simply the percentage of times the best system is correctly identified (i.e. ranked highest when the system selection algorithm is terminated) over multiple runs of the selection algorithm on random

ability in (TP,FP,TN,FN) counts, we had to use a Dirichlet-Compound-Multinomial with shared Dirichlet prior rather than a simple Multinomial with a shared Dirichlet Prior.

[2]We make use of the JAGS (Just Another Gibbs Sampler) toolkit (Plummer, 2003) for inference in our model.

reorderings of the document collection. We emphasise that, in these experiments, we simulate a scenario where the aim is to select the best system with the minimum number of queries to showcase the effectiveness of our approach.

4.1 Baselines

As baselines, we consider the minimum number of documents needed by the standard statistical power approach. The power of a binary hypothesis testing is the probability that the test correctly rejects the null hypothesis (H_0) when the alternative hypothesis (H_1) is true. In order to find a lower-bound for the number of documents, we make use of the power calculation for a paired T-Test.

The T-Test indicates whether or not the difference between two groups' averages most likely reflects a "real" difference in the population from which the groups were sampled. Assuming we have two competing systems, we can set up a T-Test to assess whether there is a meaningful difference between the F-measures of the two systems.

We assume an efficient experimental design where the same number of (identical) documents are sent to each system. Assuming a typical power setting of 80% and a significance level of 5%, we can calculate an "Oracle baseline" by making use of the true effect size (the standardised difference in mean performance) across the top two systems.[3] Obviously this quantity would not be known a-priori of running the experiment, hence the sample size calculated based on this effect size provides a lower-bound on the number of samples that ought be needed[4].

Across the systems, average performance on individual documents will vary due to variations in the inherent difficulty of each document. In other words, some documents are harder to label than others. Thus we make use of a paired sample test for the power calculation. Effect sizes are calculated as follows:

- For the synthetic experiments, the variation in difficulty of the documents is not modelled, so we calculate the effect size by simply using the parameters of the simulation as: $\frac{\mu_1 - \mu 2}{\sqrt{(\sigma_1^2 + \sigma_2^2)}}$, where μ_1 is the mean performance on the best system and μ_2 is the mean performance on the second best system (likewise for the standard deviations σ_1 and σ_2.
- For the experiments on real data, variation will be document dependent and hence we calculate the effect size as $\frac{AVG(f_1 - f_2)}{STDEV(f_1 - f_2)}$, where we directly measure the average and standard deviation of the performance differences between the top performing APIs across the documents.

Since we are comparing many APIs at once, and a priori of running the experiment we don't know which two systems are the best, we make use of two settings for the confidence level (aka P-value threshold) for the power calculation:

- **Baseline$_2$:** Assume the top two systems are known a priori and use the significance level of 5% directly to produce a lower bound on the number of iterations.
- **Baseline$_{K-1}$:** Assume one system is new and is being compared with all other $k-1$ APIs. We reduce the required significance level α using a Bonferroni correction to be $\alpha/(k-1)$ to take into account the $k-1$ comparisons being performed.

We stress that this is an unrealistic scenario in which the effect size is known before running the experiment. If this value is not known or needs to be estimated before the experiment a much larger value would be used, for example a value based on the error threshold ϵ might be appropriate.

4.2 Synthetic Data Experiments

Datasets. For the synthetic data, we generate the (TP, FP, TN, FN) counts of applying the competing systems on hypothetical documents, assuming that we know the systems' *true* rates. An important factor in the difficulty of the problem is the different in the Fscore of the top two performing systems, which we denote by *margin*. We consider three levels of problem difficulty by considering the margin $m \in \{.01, .025, .05\}$, and for each margin we consider 5 configurations whose competing systems have the specified margin. Having the true $(\hat{TP}, \hat{FP}, \hat{TN}, \hat{FN})$ rates for a competing system, the count statistics for its results on hypothetical documents are

[3]For the power calculation, we use the following R command `power.t.test(delta=effect, sd=1, sig.level=0.05, power=.8, type="paired", alternative="one.sided")`.

[4]Note that the T-Test assumes Gaussian distributed data, but the violation of this assumption is unlikely to greatly effect the sample size estimates.

	margin .05		margin .025		margin .01	
	# queries	success%	# queries	success%	# queries	success%
Baseline$_2$	120	-	440	-	2725	-
Baseline$_{K-1}$	185	-	680	-	4195	-
Hierarchical						
+ Thomp. Samp.	22 ± 5	100	41 ± 6	96	66 ± 8	100
+ Pure Eexpl. TS	19 ± 3	100	27 ± 5	100	64 ± 9	96
Gaussian						
+ Thomp. Samp.	513 ± 46	100	1169 ± 84	100	2000 ± 0	100
+ Pure Eexpl. TS	360 ± 33	100	848 ± 63	100	1965 ± 19	100

Table 1: Average number of queries across different margins. The number of systems is 5, and the maximum number of queries is set to 2000, and $\delta = 0.05$.

then generated based on our generative model. For each competing configuration, we repeat the experiment multiple times in order to account for the randomness inherent in the algorithms and the generated documents. In different experiments, we let the number of competing systems K be $\{5, 10, 20\}$.

Margin and the number of systems. In this experiment, we investigate the relation between the margin and the number of documents queried by each algorithm. Intuitively, as the margin between the top performing systems decreases, more queries are required to segregate the best system among the top performing ones. We run each algorithm for 500 times on the competing configurations for each margin with $K = 5$. The maximum number of queries allowed is 2000, and the algorithm can terminate earlier as soon as $\alpha_{T,a} \geq .95$, i.e. $\delta = 0.05$.

Table 1 summarises the average number of queries and the success rates of TS and PETS in combination with our hierarchical Bayesian model for F-measure across different margins. We see that the number of query documents increases as the margin decreases. It is also worth noting that PETS requires slightly smaller number of queries than Thompson Sampling. Interestingly, the number of samples required by the hypothesis testing baselines is much more than that required by the TS/PETS combined with our hierarchical model.

We then ask the question whether the number of competing systems is important. Table 2 summarises the average number of queries and the success rate of each algorithm on the competing configurations for the margin 0.05 for varying number of systems $K \in \{5, 10, 20\}$. As seen, the number of queries increases (sub)linearly with the number of competing systems.

Hierarchical vs Gaussian. We compare our hierarchical model for capturing the uncertainty over F-measure with the Gaussian distribution. That is, we associate a Gaussian distribution to each system to model its posterior over the F-measure. The use of the Gaussian distribution to model the mean of sampled F-measures is motivated by the law of large numbers. This approach directly models the uncertainty of a system's F-measure, as opposed to our indirect modelling approach where posterior distribution is constructed using the distribution of $(\hat{FP}, \hat{FN}, \hat{TP}, \hat{TN})$ rates.

Tables 1 and 2 show the average number of queries and success rates for algorithms using our hierarchical model vs the Gaussian distribution based model. The general trend is that using the Gaussian model in TS/PETS requires significantly more queries compared to the hierarchical model as well as the baselines. Needing more queries compared to the baselines highlights the importance of choosing the right distribution for capturing the uncertainty over the F-measure in TS/PETS. Needing more queries compared to the hierarchical variant is somewhat expected as the synthetic data is generated according to the hierarchical model. However, we will see similar trends in the experiments on the real data.

4.3 Sentiment Analysis

We consider the task of sentence level sentiment prediction for medical documents. The aim is to benchmark systems according to how well they can predict the polarity of sentences contained in a medical report, where each report corresponds

	$K:$ 5		$K:$ 10		$K:$ 20	
	# queries	success%	# queries	success%	# queries	success%
Baseline$_2$	120	-	245	-	400	-
Baseline$_{K-1}$	185	-	380	-	620	-
Hierarchical						
+ Thomp. Samp.	22 ± 5	100	43 ± 6	96	70 ± 14	100
+ Pure Eexpl. TS	19 ± 3	100	35 ± 5	100	64 ± 13	100
Gaussian						
+ Thomp. Samp.	513 ± 46	100	1019 ± 59	100	1404 ± 75	100
+ Pure Eexpl. TS	360 ± 33	100	661 ± 38	100	937 ± 62	100

Table 2: Average number of queries across different number of competing systems. The margin is 0.05, and the maximum number of queries is set to 2000, and $\delta = 0.05$.

to a patient.

Dataset. We make use of a biomedical corpus (Martinez et al., 2015) consisting of CT reports for fungal disease detection collected from three hospitals. For each report, only the free text section were used, which contains the radiologist's understanding of the scan and the reason for the requested scan as written by clinicians. Every report was de-identified: any potentially identifying information such as name, address, age/birthday, gender were removed. There are a total of 358 test documents, where the average number of sentences per document is 23.

Competing Systems. We make use of a variant of the coarse-to-fine model proposed in (McDonald et al., 2007) for sentiment analysis. Briefly speaking, the model couples the sentiment of the sentences contained in a report with the overall sentiment of the report. We train four versions of the model, each of which corresponds to a different training condition:

- M_{full}: where the model is trained on the fully annotated data D_F, i.e. the data annotated at both the sentence and report level.
- M_{partial}: where the model is trained on both D_F and the partially annotated data D_P in which the sentence level annotation is missing but the reports are labeled.
- M_{unlab}: where the model is trained on D_F and D_U in which the annotation is missing at both sentence and report level.
- M_{all}: where the model is trained on all of the available data described above.

	# queries	% success
Baseline$_2$	856	-
Baseline$_{K-1}$	1320	-
Hierarchical		
+ Thomp. Samp.	152 ± 36	100
+ Pure Eexpl. TS	123 ± 37	100
Gaussian		
+ Thomp. Samp.	500 ± 0	100
+ Pure Eexpl. TS	500 ± 0	100

Table 3: Sentiment classification for biomedical reports with 4 competing models. The maximum number of queries is set to 500, and $\delta = 0.05$.

We expect M_{all} to outperform the other models. The aim is to analyse the behaviour of our best system selection methods on real data compared to the baselines.

Results. Table 3 presents the results. As seen the number of queries needed for the TS/PETS combined with the hierarchical model is much less than that of the baselines and the Gaussian variant.

4.4 Named Entity Recognition

In our second set of experiment, we attempt to see how our frameworks and F_1 models perform using realistic data.

MASC Corpus. For benchmarking the NER systems, we use the Manually Annotated Sub-Corpus (MASC) (Ide et al., 2008) that includes 19 different domains. The corpus consists of approximately 500K words of contemporary American English written and spoken data drawn from the

Open American National Corpus (OANC). This corpus includes a wide variety of linguistic annotations with a balanced selection of texts from a broad range of genres/domains. The diversity of the corpus will enable us to assess the robustness of tools across different domains. The number of documents in the MASC corpus is about 392.

Competing Systems. We evaluate the performance of 5 popular NER systems implemented as API in third party implementations:

- OpenNLP (Ingersoll et al., 2013): The Apache OpenNLP library is a machine learning based toolkit for the text processing. It is based on the maximum entropy and perceptron.

- Stanford NER (Finkel et al., 2005): It is based on linear chain Conditional Random Field (CRF) sequence models. It is part of the Stanford CoreNLP, which is an integrated suite of NLP tools in Java.

- ANNIE (Cunningham et al., 2002): ANNIE uses gazetteer-based lookups and finite state machines for entity identification and classification. It can recognise persons, locations, organisations, dates, addresses and other named entity types. ANNIE is part of the GATE framework. It can be used as a Web Service but it also provides its own interface for independent use.

- Meaning Cloud (MeaningCloud-LLC, 1998): It is based on a hybrid approach combining machine learning with a rule based system. The software is available as a cloud based solution and on-premise as a plugin module for the GATE framework.

- LingPipe (Alias-i, 2008): It is a set of Java libraries developed by Alias-I for NLP. The NER component is based on a 1st-order Hidden Markov Model with variable-length n-grams as the feature set and uses the IOB annotation scheme for the output.

Results. Table 4 presents the results. As seen the number of queries needed for the TS/PETS combined with the hierarchical model is much less than that of the baselines and the Gaussian variant.

	# queries	% success
Baseline$_2$	125	-
Baseline$_{K-1}$	240	-
Hierarchical		
+ Thomp. Samp.	25 ± 6	99
+ Pure Eexpl. TS	24 ± 5	98
Gaussian		
+ Thomp. Samp.	539 ± 3	100
+ Pure Eexpl. TS	412 ± 2	100

Table 4: Named entity recognition on MASC documents with 5 competing systems. The maximum number of queries is set to 2000, and $\delta = .05$.

5 Conclusion

We have presented a novel approach for benchmarking NLP systems based on the multi-armed bandit (MAB) problem. We have proposed a hierarchical generative model to represent the uncertainty in the performance measures of the competing systems, to be used by the Thompson Sampling algorithm to solve the resulting MAB problem. Experimental results on both synthetic and real data show that our approach requires significantly fewer queries compared to the standard benchmarking technique to identify the best system according to F-measure. Future work includes applying our approach to other NLP problems, particularly emerging document-level problem settings such document-wise machine translation.

Acknowledgments

G. H. appreciates fruitful discussions with Mohammad Ghavamzadeh and Yasin Abbasi-Yadkori. We are grateful to reviewers for their insightful feedback and comments.

References

Alias-i. 2008. Lingpipe. Available at: http://alias-i.com/lingpipe.

Sébastien Bubeck, Rémi Munos, and Gilles Stoltz. 2009. Pure exploration in multi-armed bandits problems. In *International conference on Algorithmic learning theory*, pages 23–37. Springer.

Olivier Chapelle and Lihong Li. 2011. An empirical evaluation of thompson sampling. In *Advances in neural information processing systems*, pages 2249–2257.

Hamish Cunningham, Diana Maynard, Kalina Bontcheva, and Valentin Tablan. 2002. GATE: A Framework and Graphical Development Environment for Robust NLP Tools and Applications. In *Proceedings of the 40th Anniversary Meeting of the Association for Computational Linguistics (ACL'02)*.

Janez Demšar. 2006. Statistical comparisons of classifiers over multiple data sets. *J. Mach. Learn. Res.*, 7:1–30, December.

Jenny Rose Finkel, Trond Grenager, and Christopher Manning. 2005. Incorporating non-local information into information extraction systems by gibbs sampling. In *Proceedings of the 43rd Annual Meeting on Association for Computational Linguistics*, pages 363–370. Association for Computational Linguistics.

Victor Gabillon, Mohammad Ghavamzadeh, and Alessandro Lazaric. 2012. Best arm identification: A unified approach to fixed budget and fixed confidence. In *Advances in Neural Information Processing Systems*, pages 3212–3220.

Nancy Ide, Collin Baker, Christiane Fellbaum, and Charles Fillmore. 2008. Masc: The manually annotated sub-corpus of american english. In *In Proceedings of the Sixth International Conference on Language Resources and Evaluation (LREC*. Citeseer.

Grant S Ingersoll, Thomas S Morton, and Andrew L Farris. 2013. *Taming text: how to find, organize, and manipulate it*. Manning Publications Co.

Oded Maron and Andrew W Moore. 1993. Hoeffding races: Accelerating model selection search for classification and function approximation. *Robotics Institute*, page 263.

Ryan McDonald, Kerry Hannan, Tyler Neylon, Mike Wells, and Jeff Reynar. 2007. Structured models for fine-to-coarse sentiment analysis. In *Proceedings of the 45th Annual Meeting of the Association of Computational Linguistics*, pages 432–439, Prague, Czech Republic, June. Association for Computational Linguistics.

MeaningCloud-LLC. 1998. Meaning Cloud. Available at: https://www.meaningcloud.com/.

Martyn Plummer. 2003. JAGS: A program for analysis of Bayesian graphical models using Gibbs sampling. In *Proceedings of the 3rd International Workshop on Distributed Statistical Computing*.

Daniel Russo. 2016. Simple bayesian algorithms for best arm identification. *arXiv preprint arXiv:1602.08448*.

Steven L Scott. 2015. Multi-armed bandit experiments in the online service economy. *Applied Stochastic Models in Business and Industry*, 31(1):37–45.

W.R. Thompson. 1933. On the likelihood that one unknown probability exceeds another in view of the evidence of two samples. *Biometrika*, 25(3–4):285––294.