A Bayesian Approach for Sequence Tagging with Crowds

Edwin Simpson and Iryna Gurevych
Ubiquitous Knowledge Processing Lab
Department of Computer Science
Technische Universität Darmstadt
https://www.ukp.tu-darmstadt.de
{simpson,gurevych}@ukp.informatik.tu-darmstadt.de

Abstract

Current methods for sequence tagging, a core task in NLP, are data hungry, which motivates the use of crowdsourcing as a cheap way to obtain labelled data. However, annotators are often unreliable and current aggregation methods cannot capture common types of span annotation errors. To address this, we propose a Bayesian method for aggregating sequence tags that reduces errors by modelling sequential dependencies between the annotations as well as the ground-truth labels. By taking a Bayesian approach, we account for uncertainty in the model due to both annotator errors and the lack of data for modelling annotators who complete few tasks. We evaluate our model on crowdsourced data for named entity recognition, information extraction and argument mining, showing that our sequential model outperforms the previous state of the art. We also find that our approach can reduce crowdsourcing costs through more effective active learning, as it better captures uncertainty in the sequence labels when there are few annotations.

1 Introduction

Current methods for *sequence tagging*, a core task in NLP, use deep neural networks that require tens of thousands of labelled documents for training (Ma and Hovy, 2016; Lample et al., 2016). This presents a challenge when facing new domains or tasks, where obtaining labels is often time-consuming or costly. Labelled data can be obtained cheaply by crowdsourcing, in which large numbers of untrained workers annotate documents instead of more expensive experts. For sequence tagging, this results in multiple sequences of unreliable labels for each document.

Probabilistic methods for aggregating crowdsourced data have been shown to be more accurate than simple heuristics such as majority voting (Raykar et al., 2010; Sheshadri and Lease, 2013; Rodrigues et al., 2013; Hovy et al., 2013). However, existing methods for aggregating sequence labels cannot model dependencies between the annotators' labels (Rodrigues et al., 2014; Nguyen et al., 2017) and hence do not account for their effect on annotator noise and bias. In this paper, we remedy this by proposing a sequential annotator model and applying it to tasks that follow a *beginning, inside, outside* (*BIO*) scheme, in which the first token in a span of type 'x' is labelled 'B-x', subsequent tokens are labelled 'I-x', and tokens outside spans are labelled 'O'.

When learning from noisy or small datasets, commonly-used methods based on maximum likelihood estimation may produce over-confident predictions (Xiong et al., 2011; Srivastava et al., 2014). In contrast, Bayesian inference accounts for model uncertainty when making predictions. Unlike alternative methods that optimize the values for model parameters, Bayesian inference integrates over all possible values of a parameter, weighted by a prior distribution that captures background knowledge. The resulting posterior probabilities improve downstream decision making as they include the probability of errors due to a lack of knowledge. For example, during active learning, posterior probabilities assist with selecting the most informative data points (Settles, 2010).

In this paper, we develop *Bayesian sequence combination (BSC)*, building on prior work that has demonstrated the advantages of Bayesian inference for aggregating unreliable classifications (Kim and Ghahramani, 2012; Simpson et al., 2013; Felt et al., 2016; Paun et al., 2018). BSC is the first fully-Bayesian method for aggregating sequence labels from multiple annotators. As a core component of BSC, we also introduce the *sequential confusion matrix (seq)*, a probabilistic model

Proceedings of the 2019 Conference on Empirical Methods in Natural Language Processing and the 9th International Joint Conference on Natural Language Processing, pages 1093–1104,
Hong Kong, China, November 3–7, 2019.

of annotator noise and bias, which goes beyond previous work by modelling sequential dependencies between annotators' labels. Further contributions include a theoretical comparison of the probabilistic models of annotator noise and bias, and an empirical evaluation on three sequence labelling tasks, in which *BSC* with *seq* consistently outperforms the previous state of the art. We make all of our code and data freely available[1].

2 Related Work

Sheshadri and Lease (2013) benchmarked several aggregation models for non-sequential classifications, obtaining the most consistent performance from that of Raykar et al. (2010), who model the reliability of individual annotators using probabilistic confusion matrices, as proposed by Dawid and Skene (1979). Simpson et al. (2013) showed that a Bayesian variant of Dawid and Skene's model can outperform maximum likelihood approaches and simple heuristics when combining crowds of image annotators. This Bayesian variant, independent Bayesian classifier combination (*IBCC*) (Kim and Ghahramani, 2012), was originally used to combine ensembles of automated classifiers rather than human annotators. While traditional ensemble methods such as boosting focus on how to generate new classifiers (Dietterich, 2000), IBCC is concerned with modelling the reliability of each classifier in a given set of classifiers. To reduce the number of parameters in multi-class problems, Hovy et al. (2013) proposed *MACE*, and showed that it performed better under a Bayesian treatment on NLP tasks. Paun et al. (2018) further illustrated the advantages of Bayesian models of annotator ability on NLP classification tasks with different levels of annotation sparsity and noise.

We expand this previous work by detailing the relationships between several annotator models and extending them to sequential classification. Here we focus on the core annotator representation, rather than extensions for clustering annotators (Venanzi et al., 2014; Moreno et al., 2015), modeling their dynamics (Simpson et al., 2013), adapting to task difficulty (Whitehill et al., 2009; Bachrach et al., 2012), or time spent by annotators (Venanzi et al., 2016).

Methods for aggregating sequence labels include *CRF-MA* (Rodrigues et al., 2014), a CRF-based model that assumes only one annotator is correct for any given label. Recently, Nguyen et al. (2017) proposed a hidden Markov model (HMM) approach that outperformed CRF-MA, called *HMM-crowd*. Both CRF-MA and HMM-crowd use simpler annotator models than Dawid and Skene (1979) that do not capture the effect of sequential dependencies on annotator reliability. Neither CRF-MA nor HMM-crowd use a fully Bayesian approach, which has been shown to be more effective for handling uncertainty due to noise in crowdsourced data for non-sequential classification (Kim and Ghahramani, 2012; Simpson et al., 2013; Venanzi et al., 2014; Moreno et al., 2015). In this paper, we develop a sequential annotator model and an approximate Bayesian method for aggregating sequence labels.

3 Modeling Sequential Annotators

When combining multiple annotators with varying skill levels, we can improve performance by modelling their individual noise and bias using a probabilistic model. Here, we describe several models that do not consider dependencies between annotations in a sequence, before defining *seq*, a new extension that captures sequential dependencies. Probabilistic annotator models each define a different function, A, for the likelihood that the annotator chooses label c_τ given the true label t_τ, for the τth token in a sequence.

Accuracy model (acc): the basis of several previous methods (Donmez et al., 2010; Rodrigues et al., 2013), *acc* uses a single parameter for each annotator's accuracy, π:

$$A = p(c_\tau = i | t_\tau = j, \pi) = \begin{Bmatrix} \pi & \text{where } i = j \\ \frac{1-\pi}{J-1} & \text{otherwise} \end{Bmatrix}, \quad (1)$$

where J is the number of classes. This may be unsuitable when one class label dominates the data, since a spammer who always selects the most common label will nonetheless have a high π.

Spamming model (spam): proposed as part of MACE (Hovy et al., 2013), this model also assumes constant accuracy, π, but that when an annotator is incorrect, they label according to a spamming distribution, $\boldsymbol{\xi}$, that is independent of

[1] http://github.com/ukplab/arxiv2018-bayesian-ensembles

the true label, t_τ.

$$A = p(c_\tau = i | t_\tau = j, \pi, \boldsymbol{\xi}) = \begin{cases} \pi + (1-\pi)\xi_j & \text{where } i = j \\ (1-\pi)\xi_j & \text{otherwise} \end{cases}. \quad (2)$$

This addresses the case where spammers choose the dominant label but does not explicitly model different error rates in each class. For example, if an annotator is better at detecting type 'x' spans than type 'y', or if they frequently miss the first token in a span, thereby labelling the start of a span as 'O' when the true label is 'B-x', this would not be explicitly modelled by *spam*.

Confusion vector (CV): this approach learns a separate accuracy for each class label (Nguyen et al., 2017) using parameter vector, $\boldsymbol{\pi}$, of size J:

$$A = p(c_\tau = i | t_\tau = j, \boldsymbol{\pi}) = \begin{cases} \pi_j & \text{where } i = j \\ \frac{1-\pi_j}{J-1} & \text{otherwise} \end{cases}. \quad (3)$$

This model does not capture spamming patterns where one of the incorrect labels has a much higher likelihood than the others.

Confusion matrix (CM) (Dawid and Skene, 1979): this model can be seen as an expansion of the confusion vector so that $\boldsymbol{\pi}$ becomes a $J \times J$ matrix with values given by:

$$A = p(c_\tau = i | t_\tau = j, \boldsymbol{\pi}) = \pi_{j,i}. \quad (4)$$

This requires a larger number of parameters, J^2, compared to the $J + 1$ parameters of MACE or J parameters of the confusion vector. Like *spam*, *CM* can model spammers who frequently chose one label regardless of the ground truth, but also models different error rates and biases for each class. However, *CM* ignores dependencies between annotations in a sequence, such as the fact that an 'I' cannot immediately follow an 'O'.

Sequential Confusion Matrix (seq): we introduce a new extension to the confusion matrix to model the dependency of each label in a sequence on its predecessor, giving the following likelihood:

$$A = p(c_\tau = i | c_{\tau-1} = \iota, t_\tau = j, \boldsymbol{\pi}) = \pi_{j,\iota,i}, \quad (5)$$

where $\boldsymbol{\pi}$ is now three-dimensional with size $J \times J \times J$. In the case of disallowed transitions, e.g. from $c_{\tau-1}$ ='O' to c_τ ='I', the value $\pi_{j,c_{\tau-1},c_\tau} \approx 0$, $\forall j$ is fixed *a priori*. The sequential model can capture phenomena such as a tendency toward overly long sequences, by learning that I is more likely to follow another I, so that $\pi_{O,I,I} > \pi_{O,I,O}$. A tendency to split spans by inserting 'B' in place of 'I' can be modelled by increasing the value of $\pi_{I,I,B}$ without affecting $\pi_{I,B,B}$ and $\pi_{I,O,B}$.

The annotator models presented in this section include the most widespread models for NLP annotation tasks, and can be seen as extensions of one another. The choice of annotator model for a particular annotator depends on the developer's understanding of the annotation task: if the annotations have sequential dependencies, this suggests the *seq* model; for non-sequential classifications *CM* may be effective with small (≤ 5) numbers of classes; *spam* may be more suitable if there are many classes, as the number of parameters to learn is low. However, there is also a trade-off between the expressiveness of the model and the number of parameters that must be learned. Simpler models with fewer parameters may be effective if there are only small numbers of annotations from each annotator. The next section shows how these annotator models can be used as components of a complete model for aggregating sequential annotations.

4 A Generative Model for Bayesian Sequence Combination

To construct a generative model for *Bayesian sequence combination (BSC)*, we first define a hidden Markov model (HMM) with states $t_{n,\tau}$ and observations $x_{n,\tau}$ using categorical distributions:

$$t_{n,\tau} \sim \text{Cat}(\boldsymbol{T}_{t_{n,\tau-1}}), \quad (6)$$

$$x_{n,\tau} \sim \text{Cat}(\boldsymbol{\rho}_{t_{n,\tau}}), \quad (7)$$

where $\boldsymbol{T}_j$ is a row of a transition matrix $\boldsymbol{T}$, and $\boldsymbol{\rho}_j$ is a vector of observation likelihoods for state j. For text tagging, n indicates a document and τ a token index, while each state $t_{n,\tau}$ is a true sequence label and $x_{n,\tau}$ is a token. To provide a Bayesian treatment, we assume that $\boldsymbol{T}_j$ and $\boldsymbol{\rho}_j$ have Dirichlet distribution priors as follows:

$$T_j \sim \text{Dir}(\boldsymbol{\gamma}_j), \qquad \boldsymbol{\rho}_j \sim \text{Dir}(\boldsymbol{\kappa}_j), \quad (8)$$

where $\boldsymbol{\gamma}_j$ and $\boldsymbol{\kappa}_j$ are hyperparameters.

Next, we assume one of the annotator models described in Section 3 for each of K annotators. Selecting an annotator model is a design choice, and all can be coupled with the Bayesian HMM above to form a complete BSC model. In

our experiments in Section 6, we compare different choices of annotator model as components of BSC. All the parameters of these annotator models are probabilities, so to provide a Bayesian treatment, we assume that they have Dirichlet priors. For annotator k's annotator model, we refer to the hyperparameters of its Dirichlet prior as $\boldsymbol{\alpha}^{(k)}$. The annotator model defines a categorical likelihood over each annotation, $\boldsymbol{c}_{n,\tau}^{(k)}$:

$$c_{n,\tau}^{(k)} \sim \text{Cat}([A^{(k)}(t_{n,\tau}, 1, \boldsymbol{c}_{n,\tau-1}^{(k)}), ..., A^{(k)}(t_{n,\tau}, J, \boldsymbol{c}_{n,\tau-1}^{(k)})]). \quad (9)$$

The annotators are assumed to be conditionally independent of one another given the true labels, $\boldsymbol{t}$, which means that their errors are assumed to be uncorrelated. This is a strong assumption when considering that the annotators have to make their decisions based on the same input data. However, in practice, dependencies do not usually cause the most probable label to change (Zhang, 2004), hence the performance of classifier combination methods is only slightly degraded, while avoiding the complexity of modelling dependencies between annotators (Kim and Ghahramani, 2012).

Joint distribution: the complete model can be represented by the joint distribution, given by:

$$p(\boldsymbol{t}, \boldsymbol{A}, \boldsymbol{T}, \boldsymbol{\rho}, \boldsymbol{c}, \boldsymbol{x} | \boldsymbol{\alpha}^{(1)}, ..., \boldsymbol{\alpha}^{(K)}, \boldsymbol{\gamma}, \boldsymbol{\kappa}) \quad (10)$$

$$= \prod_{k=1}^{K} \left\{ p(A^{(k)}|\boldsymbol{\alpha}^{(k)}) \prod_{n=1}^{N} p(\boldsymbol{c}_n^{(k)}|A^{(k)}, \boldsymbol{t}) \right\}$$

$$\prod_{n=1}^{N} \prod_{\tau=1}^{L_n} p(t_{n,\tau}|\boldsymbol{T}_{t_{n,\tau-1}}) p(x_{n,\tau}|t_{n,\tau}, \boldsymbol{\rho}_{t_{n,\tau}})$$

$$\prod_{j=1}^{J} p(\boldsymbol{T}_j|\boldsymbol{\gamma}_j) p(\boldsymbol{\rho}_j|\boldsymbol{\kappa}_j) \quad (11)$$

where $\boldsymbol{c}$ is the set of annotations for all documents from all annotators, $\boldsymbol{t}$ is the set of all sequence labels for all documents, N is the number of documents, L_n is the length of the nth document, J is the number of classes, $\boldsymbol{x}$ is the set of all word sequences for all documents and $\boldsymbol{\rho}$, $\boldsymbol{\gamma}$ and $\boldsymbol{\kappa}$ are the sets of parameters for all label classes.

5 Inference using Variational Bayes

Given a set of annotations, $\boldsymbol{c}$, we obtain a posterior distribution over sequence labels, $\boldsymbol{t}$, using *variational Bayes* (*VB*) (Attias, 2000). Unlike maximum likelihood methods such as standard expectation maximization (EM), VB considers prior distributions and accounts for parameter uncertainty due to noisy or sparse data. In comparison to other Bayesian approaches such as Markov chain Monte Carlo (MCMC), VB is often faster, readily allows incremental learning, and provides easier ways to determine convergence (Bishop, 2006). It has been successfully applied to a wide range of methods, including being used as the standard learning procedure for LDA (Blei et al., 2003), and to combining non-sequential crowdsourced classifications (Simpson et al., 2013).

The trade-off is that we must approximate the posterior distribution with an approximation that factorises between subsets of latent variables:

$$p(\boldsymbol{t}, \boldsymbol{A}, \boldsymbol{T}, \boldsymbol{\rho} | \boldsymbol{c}, \boldsymbol{x}, \boldsymbol{\alpha}, \boldsymbol{\gamma}, \boldsymbol{\kappa})$$

$$\approx \prod_{k=1}^{K} q(A^{(k)}) \prod_{j=1}^{J} \{q(\boldsymbol{T}_j) q(\boldsymbol{\rho}_j)\} \prod_{n=1}^{N} q(\boldsymbol{t}_n). \quad (12)$$

VB performs approximate inference by updating each variational factor, $q(z)$, in turn, optimising the approximate posterior distribution until it converges. Details of the theory are beyond the scope of this paper, but are explained by Bishop (2006). The VB algorithm is described in Algorithm 1, making use of update equations for the variational factors given below.

Input: Annotations $\boldsymbol{c}$, tokens $\boldsymbol{x}$

Compute initial values of $\mathbb{E} \ln A^{(k)}, \forall k$, $\mathbb{E} \ln \rho_j, \forall j$, $\mathbb{E} \ln \boldsymbol{T}_j, \forall j$ from their prior distributions.

while not_converged$(r_{n,\tau,j}, \forall n, \forall \tau, \forall j)$ **do**

  Update $r_{j,n,\tau}$, $s_{t_{j,n,\tau-1}, t_{\iota,n,\tau}}, \forall j, \forall \tau, \forall n, \forall \iota$, using forward-backward algorithm given $\boldsymbol{x}$, $\boldsymbol{c}$, $\mathbb{E} \ln \boldsymbol{T}_j, \forall j$, $\mathbb{E} \ln \boldsymbol{\rho}_j, \forall j$, $\mathbb{E} \ln \boldsymbol{A}^{(k)}, \forall k$.

  Update $\mathbb{E} \ln A^{(k)}, \forall k$, given $\boldsymbol{c}$, $r_{j,n,\tau}$.

  Update $\mathbb{E} \ln T_{j,\iota}, \forall j, \forall \iota$, given $s_{t_{j,n,\tau-1}, t_{\iota,n,\tau}}$.

  Update $\mathbb{E} \ln \rho_j, \forall j$ given $\boldsymbol{x}$, $r_{j,n,\tau}$.

end

Output: Label posteriors, $r_{n,\tau,j}, \forall n, \forall \tau, \forall j$, most probable sequence of labels, $\hat{\boldsymbol{t}}_n, \forall n$ using Viterbi algorithm

Algorithm 1: The VB algorithm for BSC.

We compute the posterior probability of each true token label, $r_{n,\tau,j} = \mathbb{E}[p(t_{n,\tau} = j|\boldsymbol{c})]$, and

of each label transition, $s_{n,\tau,j,\iota} = \mathbb{E}[p(t_{n,\tau-1} = j, t_{n,\tau} = \iota|\boldsymbol{c})]$, using the forward-backward algorithm (Ghahramani, 2001). In the forward pass, we compute:

$$\ln r^{-}_{n,\tau,j} = \ln \sum_{\iota=1}^{J} \left\{ r^{-}_{n,\tau-1,\iota} \mathrm{e}^{\mathbb{E}\ln T_{\iota,j}} \right\} + ll_{n,\tau}(j),$$

$$ll_{n,\tau}(j) = \sum_{k=1}^{K} \mathbb{E}\ln A^{(k)}\left(j, c^{(k)}_{n,\tau}, c^{(k)}_{n,\tau-1}\right) \mathbb{E}\ln \rho_{j,x_{n,\tau}}, \tag{13}$$

and in the backward pass we compute:

$$\ln \lambda_{n,\tau,j} = \ln \sum_{\iota=1}^{J} \exp\big\{ \ln \lambda_{i,\tau+1,\iota} + \mathbb{E}\ln T_{j,\iota} + ll_{n,\tau+1}(\iota) \big\}. \tag{14}$$

Then we can calculate the posteriors as follows:

$$r_{n,\tau,j} \propto r^{-}_{n,\tau,j} \lambda_{n,\tau,j}, \tag{15}$$

$$s_{n,\tau,j,\iota} \propto r^{-}_{n,\tau-1,j} \lambda_{n,\tau,\iota} \exp\{\mathbb{E}\ln T_{j,\iota} + ll_{n,\tau}(\iota)\}. \tag{16}$$

The expectations of $\ln \boldsymbol{T}$ and $\ln \boldsymbol{\rho}$ can be computed using standard equations for a Dirichlet distribution:

$$\mathbb{E}\ln T_{j,\iota} = \Psi(N_{j,\iota} + \gamma_{j,\iota}) - \Psi\left(\sum_{i=1}^{J} (N_{j,i} + \gamma_{j,i})\right), \tag{17}$$

$$\mathbb{E}\ln \rho_{j} = \Psi(o_{j,w} + \kappa_{j,w}) - \Psi\left(\sum_{v=1}^{J} (o_{j,v} + \kappa_{j,v})\right), \tag{18}$$

where Ψ is the digamma function, $N_{j,\iota} = \sum_{n=1}^{N}\sum_{\tau=1}^{L_n} s_{n,\tau,j,\iota}$ is the expected number of times that label ι follows label j, and $o_{j,w}$ is the expected number of times that word w occurs with sequence label j. Similarly, for the *seq* annotator model, the expectation terms are:

$$\mathbb{E}\ln A^{(k)}(j,l,m) = \Psi\left(N^{(k)}_{j,l,m}\right) - \Psi\left(\sum_{m'=1}^{J}\left(N^{(k)}_{j,l,m'}\right)\right). \tag{19}$$

$$N^{(k)}_{j,l,m} = \alpha^{(k)}_{j,l,m} + \sum_{n=1}^{N}\sum_{\tau=1}^{L_n} r_{n,\tau,j}\delta_{l,c^{(k)}_{n,\tau-1}}\delta_{m,c^{(k)}_{n,\tau}}, \tag{20}$$

where δ is the Kronecker delta. For other annotator models, this equation is simplified as the values of the previous labels $c^{(k)}_{n,\tau-1}$ are ignored.

data -set	#sentences or docs total	dev	test	#annotators total	/doc	#gold spans	span length
NER	6,056	2,800	3,256	47	4.9	21,612	1.51
PICO	9,480	191	191	312	6.0	700	7.74
ARG	8,000	60	100	105	5	73	17.52

Table 1: Dataset statistics. Span lengths are means.

5.1 Most Likely Sequence Labels

The approximate posterior probabilities of the true labels, $r_{j,n,\tau}$, provide confidence estimates for the labels. However, it is often useful to compute the most probable sequence of labels, $\hat{\boldsymbol{t}}_n$, using the Viterbi algorithm (Viterbi, 1967). The most probable sequence is particularly useful because, unlike $r_{j,n,\tau}$, the sequence will be consistent with any transition constraints imposed by the priors on the transition matrix $\boldsymbol{T}$, such as preventing 'O'→'I' transitions by assigning them zero probability.

6 Experiments

Datasets. We compare BSC to alternative methods on three NLP datasets containing both crowdsourced and gold sequential annotations. *NER*, the CoNLL 2003 named-entity recognition dataset (Tjong Kim Sang and De Meulder, 2003), which contains gold labels for four named entity categories (PER, LOC, ORG, MISC), with crowdsourced labels provided by (Rodrigues et al., 2014). *PICO* (Nguyen et al., 2017), consists of medical paper abstracts that have been annotated by a crowd to indicate text spans that identify the population enrolled in a clinical trial. *ARG* (Trautmann et al., 2019) contains a mixture of argumentative and non-argumentative sentences, in which the task is to mark the spans that contain pro or con arguments for a given topic. Dataset statistics are shown in Table 1. The datasets differ in typical span length, with argument components in ARG the longest, while named entities in NER spans are often only one token long.

The gold-labelled documents are split into validation and test sets. For NER, we use the split given by Nguyen et al. (2017), while for PICO and ARG, we make random splits since the splits from previous work were not available[2].

Evaluation metrics. For NER and ARG we use the CoNLL 2003 F1-score, which considers only

[2]The data splits are available from our Github repository. Since we use different splits, our results for PICO are not identical to Nguyen et al. (2017).

	NER				PICO				ARG			
	Prec.	Rec.	F1	CEE	Prec.	Rec.	F1	CEE	Prec.	Rec.	F1	CEE
Best worker	76.4	60.1	67.3	17.1	64.8	53.2	58.5	17.0	62.7	57.5	60.0	44.20
Worst worker	55.7	26.5	35.9	31.9	50.7	52.9	51.7	41.0	25.5	19.2	21.9	70.33
MV	79.9	55.3	65.4	6.24	82.5	52.8	64.3	2.55	40.0	31.5	34.8	14.03
MACE	74.4	66.0	70.0	1.01	25.4	84.1	39.0	58.2	31.2	32.9	32.0	2.62
DS	79.0	70.4	74.4	2.80	71.3	66.3	68.7	0.44	45.6	49.3	47.4	0.97
IBCC	79.0	70.4	74.4	**0.49**	72.1	66.0	68.9	**0.27**	44.9	47.9	46.4	**0.85**
HMM-crowd	80.1	69.2	74.2	1.00	75.9	66.7	71.0	0.99	43.5	37.0	40.0	3.38
BSC-acc	**83.4**	54.3	65.7	0.96	**89.4**	45.2	60.0	1.59	36.9	32.9	34.8	6.47
BSC-spam	67.9	74.1	70.9	0.89	46.7	**84.4**	60.1	1.98	55.7	53.4	54.5	2.80
BSC-CV	83.0	64.6	72.6	0.93	74.9	67.2	71.1	0.84	37.9	34.2	36.0	4.73
BSC-CM	79.9	72.2	75.8	1.46	60.1	78.8	68.2	1.49	**56.0**	57.5	56.8	3.76
BSC-seq	80.3	**74.8**	**77.4**	0.65	70.4	75.3	**72.8**	0.53	54.4	**67.1**	**60.1**	3.26
BSC-CM-notext	74.7	69.7	72.1	1.48	62.7	74.8	68.2	1.32	55.1	58.9	57.0	2.75
BSC-CM$\backslash \boldsymbol{T}$	80.0	73.0	76.3	0.99	65.8	66.7	66.2	0.28	52.9	49.3	51.1	1.69
BSC-seq-notext	81.3	71.9	76.3	0.52	81.2	59.2	68.5	0.73	36.9	52.0	43.2	5.64
BSC-seq$\backslash \boldsymbol{T}$	64.2	44.4	52.5	0.77	51.2	70.4	59.8	1.04	0.11	0.05	0.07	6.38

Table 2: Aggregating crowdsourced labels: estimating true labels for documents labelled by the crowd.

exact span matches to be correct. Incomplete named entities are typically not useful, and for ARG, it is desirable to identify complete argumentative units that make sense on their own. For medical trial populations, partial matches still contain useful information, so for PICO we use a relaxed F1-score, as in Nguyen et al. (2017), which counts the matching fractions of spans when computing precision and recall.

We also compute the cross entropy error (*CEE*, also known as log-loss). While this is a token-level rather than span-level metric, it evaluates the quality of the probability estimates produced by aggregation methods, which are useful for tasks such as active learning, as it penalises over-confident mistakes more heavily.

Evaluated Methods. We evaluate BSC in combination with all of the annotator models described in Section 4. As well-established non-sequential baselines, we include token-level majority voting (*MV*), *MACE* (Hovy et al., 2013) which uses the *spam* annotator model, Dawid-Skene (*DS*) (Dawid and Skene, 1979), which uses the *CM* annotator model, and independent Bayesian classifier combination (*IBCC*) (Kim and Ghahramani, 2012), which is a Bayesian treatment of Dawid-Skene. We also compare against the state-of-the-art sequential *HMM-crowd* method (Nguyen et al., 2017), which uses a combination of maximum *a posteriori* (or smoothed maximum likelihood) estimates for the *CV* annotator model and variational inference for an integrated hidden Markov model (HMM). HMM-Crowd and DS use non-Bayesian inference steps and can be compared with their Bayesian variants, BSC-CV and IBCC, respectively.

Besides the annotator models, BSC also makes use of text features and a transition matrix, $\boldsymbol{T}$, over true labels. We test the effect of these components by running BSC-CM and BSC-seq with no text features (*notext*), and without the transition matrix, which is replaced by simple independent class probabilities (labelled $\backslash \boldsymbol{T}$).

We tune the hyperparameters using grid search on the development sets. To limit the number of hyperparameters to tune, we optimize only three values for BSC: hyperparameters of the transition matrix, $\boldsymbol{\gamma}_j$, are set to the same value, γ_0, except for disallowed transitions, (O→I and transitions between types, e.g. I-PER→I-ORG), which are set to $1e-6$; for the annotator models, all values are set to α_0, except for disallowed transitions, which are set to $1e^{-6}$, then ϵ_0 is added to hyperparameters corresponding to correct annotations (e.g. diagonal entries in a confusion matrix). This encodes the prior assumption that annotators are more likely to have an accuracy greater than random, which avoids the non-identifiability problem in which the class labels are switched around.

Aggregation Results. This task is to combine multiple crowdsourced labels and predict the true labels. The results are shown in Table 2. BSC-seq outperforms the other approaches, including the previous state of the art, HMM-crowd (sig-

Method	Data -set	exact match	wrong type	partial match	missed span	false +ve	late start	early start	late finish	early finish	fused spans	splits	inv- alid	length error
MV	NER	2869	304	196	1775	100	96	10	15	85	17	26	81	0.04
IBCC	NER	3742	386	187	829	345	107	27	43	77	47	29	74	0.12
HMM-crowd	NER	3650	334	115	1045	210	109	22	33	89	37	23	0	0.03
BSC-CV	NER	3381	284	80	1399	121	94	17	18	90	22	8	0	0.00
BSC-CM	NER	3856	362	63	863	315	124	25	63	77	53	13	0	0.14
BSC-seq	NER	3995	353	110	686	357	84	29	25	88	28	26	0	0.09
MV	PICO	144	0	60	145	48	9	11	1	0	3	9	40	1.26
IBCC	PICO	193	0	53	103	100	14	10	0	2	3	10	19	0.45
HMM-crowd	PICO	189	0	54	106	84	13	21	0	0	5	8	0	1.99
BSC-CV	PICO	156	0	76	117	81	10	25	0	0	11	0	0	2.15
BSC-CM	PICO	216	0	50	83	157	10	19	0	0	4	17	0	2.42
BSC-seq	PICO	168	0	86	95	67	17	19	5	0	4	9	0	0.61
MV	ARG	17	0	26	14	4	9	1	0	2	0	0	9	5.27
IBCC	ARG	27	1	21	8	9	7	2	0	1	0	3	9	3.43
HMM-Crowd	ARG	20	0	23	14	4	7	2	0	2	0	0	4	4.87
BSC-CV	ARG	18	0	25	14	4	12	2	0	2	0	0	0	5.37
BSC-CM	ARG	35	1	12	9	9	7	2	0	1	1	0	0	2.11
BSC-Seq	ARG	39	3	12	3	20	6	4	0	0	1	0	0	0.46

Table 3: Counts of different types of span errors.

nificant on all datasets with $p \ll .01$ using a two-tailed Wilcoxon signed-rank test). Without the text model (BSC-seq-notext) or the transition matrix (BSC-seq$\backslash \boldsymbol{T}$), BSC-seq performance decreases heavily, while BSC-CM-notext and BSC-CM$\backslash \boldsymbol{T}$ in some cases outperform BSC-CM. This suggests that *seq*, with its greater number of parameters to learn, is most effective in combination with the transition matrix and simple text model. On the ARG dataset, the scores are close to zero for BSC-seq$\backslash \boldsymbol{T}$. Further investigation shows that this is because BSC-seq$\backslash \boldsymbol{T}$ identifies many spans with one or two incorrect labels. Since we use exact span matches to compute true and false positives, these small errors reduce the scores substantially. In particular, we find a large number of missing B tags at the start of spans and misplaced O tags that split spans in the middle.

The performance of all methods across the three datasets varies greatly. With NER, the spans are short and the task is less subjective than PICO or ARG, hence its higher F1 scores. PICO uses a relaxed F1 score, meaning its scores are only slightly lower despite being a more ambiguous task. The constitution of an argument is also ambiguous, so ARG scores are lower, particularly as we use strict span-matching to compute the F1 scores. Raising the scores may be possible in future by using expert labels as training data, i.e. as known values of $\boldsymbol{t}$, which would help to put more weight on annotators with similar labelling patterns to the experts.

We categorise the errors made by key methods and list the counts for each category in Table 3. All machine learning methods tested here reduce the number of spans that were completely missed by majority voting. Note that BSC completely removes all "invalid" spans (O→I transitions) due to the sequential model with prior hyperparameters set to zero for those transitions. For PICO and ARG, which contain longer spans, BSC-seq has lower "length error" than other methods, which is the mean difference in number of tokens between the predicted and gold spans. It also reduces the number of missing spans, although in NER and ARG that comes at the cost of producing more false positives (predicting spans where there are none). Overall, BSC-seq appears to be the best choice for identifying exact span matches and reducing missed spans.

Visualising Annotator Models. To determine whether, in practice, BSC-seq really learns distinctive confusion matrices depending on the previous labels, we plot the learned annotator models for PICO as probabilistic confusion matrices in Figure 1 (for an alternative visualisation, see Figure 3 in the appendix). We clustered the confusion matrices into five groups by applying K-means to their posterior expected values, then plotted the means for each cluster. Each column in the figure shows the confusion matrices corresponding to the same cluster of annotators. In all clusters, BSC-seq learns different confusion matrices depending on the previous label. There are also large varia-

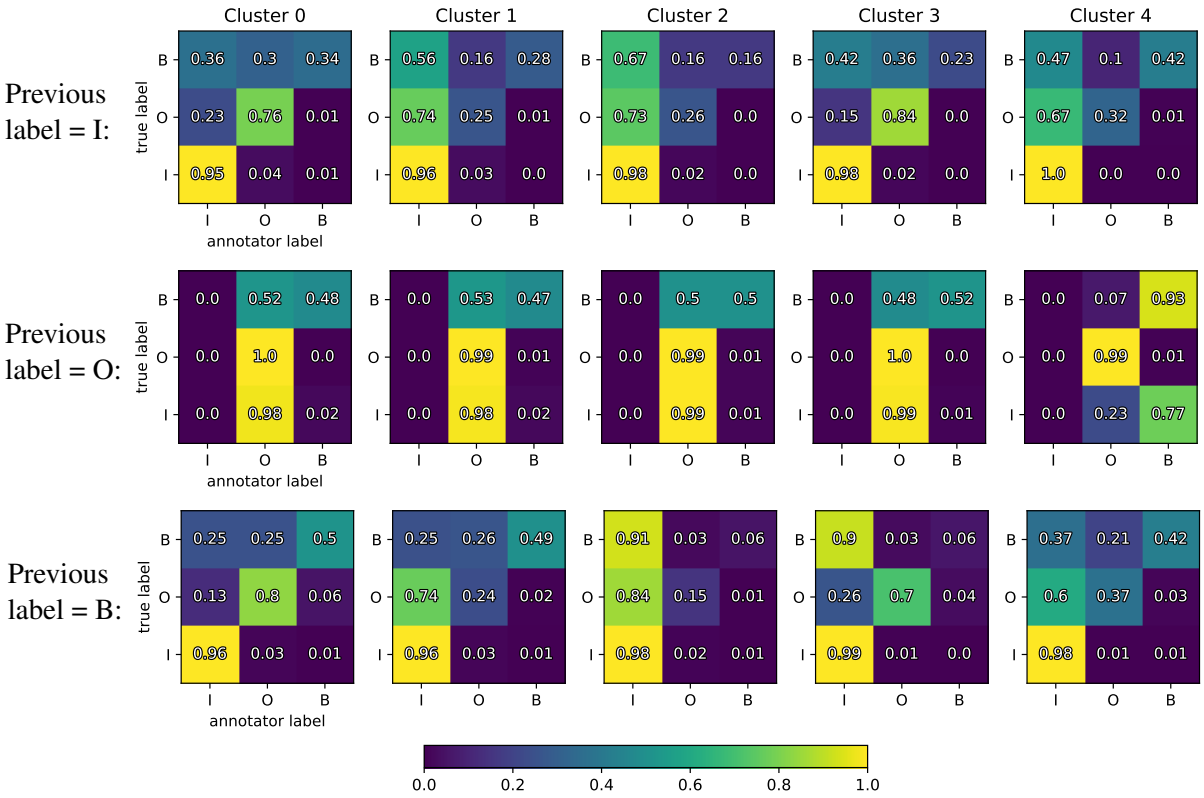

Figure 1: Clusters of annotators in the PICO dataset. Each column shows the mean of the confusion matrices in the *seq* annotator model for the members of one cluster, as inferred by BSC-seq. Each row corresponds to the mean confusion matrices conditioned on the annotator's previous label. The confusion matrices are plotted as heatmaps, where colours indicate the probabilities, which are also given by the numbers in each square.

tions between the clusters. The third column, for example, shows annotators with a tendency toward B→I transitions regardless of the true label, while other clusters indicate very different labelling behaviour. The model therefore appears able to learn distinct confusion matrices for different workers given previous labels, which supports the use of sequential annotator models.

Active Learning. Active learning is an iterative process that can reduce the amount of labelled data required to train a model. At each iteration, the active learning strategy selects informative data points, obtains their labels, then re-trains a labelling model given the new labels. The updated model is then used in the next iteration to identify the most informative data points.

We simulate active learning in a crowdsourcing scenario where the goal is to learn the true labels, $\boldsymbol{t}$, by selecting documents for the crowd to label. Each document can be labelled multiple times by different workers. In contrast, in a passive learning setup, the number of annotations per document is usually constant across all documents. For example, in the PICO dataset, each document was labelled six times. The aim of active learning is to decrease the number of annotations required by avoiding relabelling documents whose true labels can be determined with high confidence from fewer labels.

We simulate active learning using the *least confidence* strategy, shown to be effective by Settles and Craven (2008), as described in Algorithm 2. At each iteration, we estimate $\boldsymbol{t}$ from the current set of crowdsourced labels, $\boldsymbol{c}$, using one of the methods from our previous experiments as the labelling model, then use this model to determine the least confident $batch\_size$ documents to be labelled by the crowd. If the simulation has requested all the labels for a document that are available in our dataset, this document is simply ignored when choosing new batches and is not selected again.

We hypothesise that BSC will learn more quickly than non-sequential and non-Bayesian methods in the active learning simulation. Bayesian methods such as BSC account for

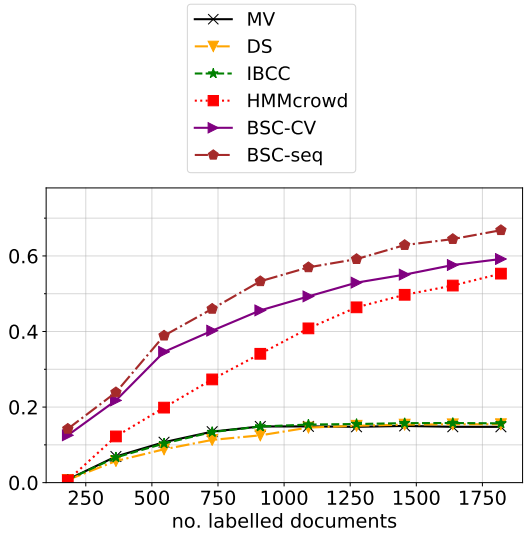

Figure 2: F1 scores for active learning simulations on NER using least-confidence uncertainty sampling.

uncertainty in the model parameters when estimating confidence, hence can be used to choose data points that rapidly improve the model itself. Sequential models such as BSC can also account for dependencies between sequence labels when estimating confidence.

Input: A random $initial\_set$ of training labels, the same for all methods.

Set training set $\boldsymbol{c} = initial\_set$

while *training set size* $< max\_no\_labels$ **do**

  Train model on $\boldsymbol{c}$

  For each document n, compute $LC_n = 1 - p(\boldsymbol{t}_n^*|\boldsymbol{c})$, where $\boldsymbol{t}_n^*$ is the probability of the most likely sequence of labels for n.

  Obtain annotations for $batch\_size$ documents with the highest values of LC (least confidence), and add them to $\boldsymbol{c}$

end

Algorithm 2: Active learning simulation using least-confidence sampling.

Figure 2 plots the mean F1 scores over ten repeats of the active learning simulation on the NER dataset (for clarity, we only plot key methods). When the number of iterations is very small, neither IBCC nor DS are able to outperform majority vote, and only produce a very small benefit as the number of labels grows. This highlights the need for a sequential model such as BSC or HMM-crowd for effective active learning with small numbers of labels. IBCC learns slightly quicker than DS, while BSC-CV clearly outperforms HMM-crowd: we believe this difference is due to the Bayesian treatment of IBCC and BSC, which means they are better able to estimate confidence than DS and HMM-crowd, which use maximum likelihood and maximum a posteriori inference. BSC-seq produces the best overall performance, and the gap grows as the number of labels increases, since more data is required to learn the more complex annotator model.

7 Conclusions

We proposed BSC, a novel Bayesian approach to aggregating sequence labels that can be combined with several different models of annotator noise and bias. To model the effect of dependencies between labels on annotator noise and bias, we introduced the *seq* annotator model. Our results demonstrated the benefits of BSC over established non-sequential methods, such as MACE, Dawid-Skene (DS), and IBCC. We also showed the advantages of a Bayesian approach for active learning, and that the combination of *BSC* with the *seq* annotator model improves the state-of-the-art over HMM-crowd on three NLP tasks with different types of span annotations.

In future work, we plan to adapt active learning methods for easy deployment on crowdsourcing platforms, and to investigate techniques for automatically selecting good hyperparameters without recourse to a development set, which is often unavailable at the start of a crowdsourcing process.

Acknowledgments

This work was supported by the German Research Foundation through the German-Israeli Project Cooperation (DIP, grant DA 1600/1-1 and grant GU 798/17-1), and by the German Research Foundation (DFG) as part of the QA-EduInf project (grant GU 798/18-1 and grant RI 803/12-1). We would like to thank all of those who developed the datasets used in this work.

References

Hagai Attias. 2000. A variational Bayesian framework for graphical models. In *Advances in Neural Information Processing Systems 12*, pages 209–215. MIT Press.

Yoram Bachrach, Tom Minka, John Guiver, and Thore Graepel. 2012. How to grade a test without knowing

the answers: a Bayesian graphical model for adaptive crowdsourcing and aptitude testing. In *Proceedings of the 29th International Coference on International Conference on Machine Learning*, pages 819–826. Omnipress.

Christopher. M. Bishop. 2006. *Pattern recognition and machine learning*, 4th edition. Information Science and Statistics. Springer.

David M. Blei, Andrew Y. Ng, and Michael I. Jordan. 2003. Latent Dirichlet allocation. *The Journal of Machine Learning Research*, 3:993–1022.

Alexander Philip Dawid and Allan M. Skene. 1979. Maximum likelihood estimation of observer error-rates using the EM algorithm. *Journal of the Royal Statistical Society. Series C (Applied Statistics)*, 28(1):20–28.

Thomas G Dietterich. 2000. Ensemble methods in Machine Learning. In *Multiple classifier systems*, pages 1–15. Springer.

Pinar Donmez, Jaime Carbonell, and Jeff Schneider. 2010. A probabilistic framework to learn from multiple annotators with time-varying accuracy. In *Proceedings of the 2010 SIAM International Conference on Data Mining*, pages 826–837. SIAM.

Paul Felt, Eric K. Ringger, and Kevin D. Seppi. 2016. Semantic annotation aggregation with conditional crowdsourcing models and word embeddings. In *International Conference on Computational Linguistics*, pages 1787–1796.

Zoubin Ghahramani. 2001. An introduction to hidden markov models and Bayesian networks. *International Journal of Pattern Recognition and Artificial Intelligence*, 15(01):9–42.

Dirk Hovy, Taylor Berg-Kirkpatrick, Ashish Vaswani, and Eduard H Hovy. 2013. Learning whom to trust with MACE. In *HLT-NAACL*, pages 1120–1130.

Hyun-chul Kim and Zoubin Ghahramani. 2012. Bayesian classifier combination. In *International Conference on Artificial Intelligence and Statistics*, pages 619–627.

Guillaume Lample, Miguel Ballesteros, Sandeep Subramanian, Kazuya Kawakami, and Chris Dyer. 2016. Neural architectures for named entity recognition. In *Proceedings of NAACL-HLT*, pages 260–270.

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)*, volume 1, pages 1064–1074.

Pablo G. Moreno, Yee Whye Teh, and Fernando Perez-Cruz. 2015. Bayesian nonparametric crowdsourcing. *Journal of Machine Learning Research*, 16:1607–1627.

An T Nguyen, Byron C Wallace, Junyi Jessy Li, Ani Nenkova, and Matthew Lease. 2017. Aggregating and predicting sequence labels from crowd annotations. In *Proceedings of the conference. Association for Computational Linguistics. Meeting*, volume 2017, page 299. NIH Public Access.

Silviu Paun, Bob Carpenter, Jon Chamberlain, Dirk Hovy, Udo Kruschwitz, and Massimo Poesio. 2018. Comparing Bayesian models of annotation. *Transactions of the Association for Computational Linguistics*, 6:571–585.

Vikas. C. Raykar, Shipeng Yu, Linda H. Zhao, Gerardo Hermosillo. Valadez, Charles Florin, Luca Bogoni, and Linda Moy. 2010. Learning from crowds. *Journal of Machine Learning Research*, 11:1297–1322.

Filipe Rodrigues, Francisco Pereira, and Bernardete Ribeiro. 2013. Learning from multiple annotators: distinguishing good from random labelers. *Pattern Recognition Letters*, 34(12):1428–1436.

Filipe Rodrigues, Francisco Pereira, and Bernardete Ribeiro. 2014. Sequence labeling with multiple annotators. *Machine learning*, 95(2):165–181.

Burr Settles. 2010. Active learning literature survey. *Computer Sciences Technical Report 1648, University of Wisconsin-Madison*, 52(55-66):11.

Burr Settles and Mark Craven. 2008. An analysis of active learning strategies for sequence labeling tasks. In *Proceedings of the conference on empirical methods in natural language processing*, pages 1070–1079. Association for Computational Linguistics.

Aashish Sheshadri and Matthew Lease. 2013. SQUARE: A benchmark for research on computing crowd consensus. In *First AAAI Conference on Human Computation and Crowdsourcing*.

Edwin Simpson, Stephen J. Roberts, Ioannis Psorakis, and Arfon Smith. 2013. Dynamic Bayesian combination of multiple imperfect classifiers. *Intelligent Systems Reference Library series*, Decision Making with Imperfect Decision Makers:1–35.

Nitish Srivastava, Geoffrey Hinton, Alex Krizhevsky, Ilya Sutskever, and Ruslan Salakhutdinov. 2014. Dropout: a simple way to prevent neural networks from overfitting. *The Journal of Machine Learning Research*, 15(1):1929–1958.

Erik F Tjong Kim Sang and Fien De Meulder. 2003. Introduction to the CoNLL-2003 shared task: Language-independent named entity recognition. In *Proceedings of the seventh conference on Natural language learning at HLT-NAACL 2003-Volume 4*, pages 142–147. Association for Computational Linguistics.

Dietrich Trautmann, Johannes Daxenberger, Christian Stab, Hinrich Schütze, and Iryna Gurevych. 2019. Robust argument unit recognition and classification. *arXiv preprint arXiv:1904.09688*.

Matteo Venanzi, John Guiver, Gabriella Kazai, Pushmeet Kohli, and Milad Shokouhi. 2014. Community-based Bayesian aggregation models for crowdsourcing. In *23rd international conference on World wide web*, pages 155–164.

Matteo Venanzi, John Guiver, Pushmeet Kohli, and Nicholas R Jennings. 2016. Time-sensitive Bayesian information aggregation for crowdsourcing systems. *Journal of Artificial Intelligence Research*, 56:517–545.

Andrew Viterbi. 1967. Error bounds for convolutional codes and an asymptotically optimum decoding algorithm. *IEEE transactions on Information Theory*, 13(2):260–269.

Jacob Whitehill, Ting-fan Wu, Jacob Bergsma, Javier R Movellan, and Paul L Ruvolo. 2009. Whose vote should count more: Optimal integration of labels from labelers of unknown expertise. In *Advances in neural information processing systems*, pages 2035–2043.

Hui Yuan Xiong, Yoseph Barash, and Brendan J Frey. 2011. Bayesian prediction of tissue-regulated splicing using rna sequence and cellular context. *Bioinformatics*, 27(18):2554–2562.

Harry Zhang. 2004. The optimality of naïve Bayes. In *Proceedings of the Seventeenth International Florida Artificial Intelligence Research Society Conference, FLAIRS 2004*. AAAI Press.

A Discussion of Variational Approximation

In Equation 12, each subset of latent variables has a variational factor of the form $\ln q(z) = \mathbb{E}[\ln p(z|\boldsymbol{c}, \neg z)]$, where $\neg z$ contains all the other latent variables excluding z. We perform approximate inference by using coordinate ascent to update each variational factor, $q(z)$, in turn, taking expectations with respect to the current estimates of the other variational factors. Each iteration reduces the KL-divergence between the true and approximate posteriors of Equation 12, and hence optimises a lower bound on the log marginal likelihood, also called the evidence lower bound or ELBO (Bishop, 2006; Attias, 2000).

Conjugate distributions: The prior distributions chosen for our generative model are conjugate to the distributions over the latent variables and model parameters, meaning that each $q(z)$ is the same type of distribution as the corresponding prior distribution defined in Section 4. The parameters of each variational distribution are computed in terms of expectations over the other subsets of variables.

B Update Equations for the Forward-Backward Algorithm

For the true labels, $\boldsymbol{t}$, the variational factor is:

$$\ln q(\boldsymbol{t}_n) = \sum_{n=1}^{N}\sum_{\tau=1}^{L_n}\sum_{k=1}^{K} \mathbb{E}\ln A^{(k)}\Big(t_{n,\tau}, c_{n,\tau}^{(k)}, c_{n,\tau-1}^{(k)}\Big) + \mathbb{E}\ln T_{t_{n,\tau-1},t_{n,\tau}} + \text{const.} \tag{21}$$

The forward-backward algorithm consists of two passes. The *forward pass* for each document, n, starts from $\tau = 1$ and computes:

$$\ln r^{-}_{n,\tau,j} = \ln \sum_{\iota=1}^{J}\left\{ r^{-}_{n,\tau-1,\iota} \mathrm{e}^{\mathbb{E}\ln T_{\iota,j}} \right\} + ll_{n,\tau}(j),$$

$$ll_{n,\tau}(j) = \sum_{k=1}^{K} \mathbb{E}\ln A^{(k)}\left(j, c_{n,\tau}^{(k)}, c_{n,\tau-1}^{(k)}\right) \tag{22}$$

where $r^{-}_{n,0,\iota} = 1$ where ι ='O' and 0 otherwise. The *backwards pass* starts from $\tau = L_n$ and scrolls backwards, computing:

$$\ln \lambda_{n,L_n,j} = 0, \qquad \ln \lambda_{n,\tau,j} = \ln \sum_{\iota=1}^{J} \exp\big\{ \ln \lambda_{i,\tau+1,\iota} + \mathbb{E}\ln T_{j,\iota} + ll_{n,\tau+1}(\iota)\big\}. \tag{23}$$

By applying Bayes' rule, we arrive at $r_{n,\tau,j}$ and $s_{n,\tau,j,\iota}$:

$$r_{n,\tau,j} = \frac{r^{-}_{n,\tau,j}\lambda_{n,\tau,j}}{\sum_{j'=1}^{J} r^{-}_{n,\tau,j'}\lambda_{n,\tau,j'}} \tag{24}$$

$$s_{n,\tau,j,\iota} = \frac{\tilde{s}_{n,\tau,j,\iota}}{\sum_{j'=1}^{J}\sum_{\iota'=1}^{J}\tilde{s}_{n,\tau,j',\iota'}} \tag{25}$$

$$\tilde{s}_{n,\tau,j,\iota} = r^{-}_{n,\tau-1,j}\lambda_{n,\tau,\iota}\exp\{\mathbb{E}\ln T_{j,\iota} + ll_{n,\tau}(\iota)\}.$$

Each row of the transition matrix has the factor:

$$\ln q(\boldsymbol{T}_j) = \ln \mathrm{Dir}\left([N_{j,\iota} + \gamma_{j,\iota}, \forall \iota \in \{1,..,J\}]\right), \tag{26}$$

where $N_{j,\iota} = \sum_{n=1}^{N}\sum_{\tau=1}^{L_n} s_{n,\tau,j,\iota}$ is the expected number of times that label ι follows label j. The forward-backward algorithm requires expectations of $\ln \boldsymbol{T}$ that can be computed using standard equations for a Dirichlet distribution:

$$\mathbb{E}\ln T_{j,\iota} = \Psi(N_{j,\iota} + \gamma_{j,\iota}) - \Psi\left(\sum_{\iota=1}^{J}(N_{j,\iota} + \gamma_{j,\iota})\right), \tag{27}$$

where Ψ is the digamma function.

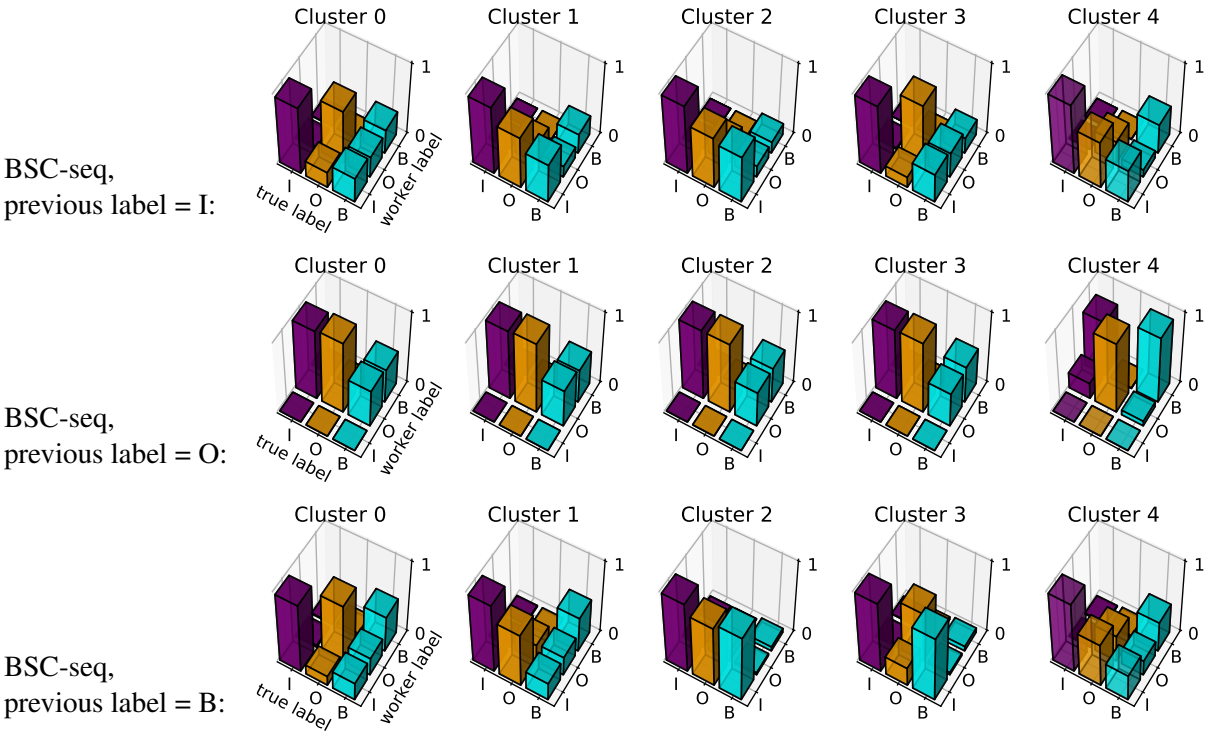

Figure 3: Clusters of annotators in PICO represented by the mean confusion matrices from BSC-seq. Heights of bars indicate likelihood of a worker (or annotator) label given the true label.

The variational factor for each annotator model is a distribution over its parameters, which differs between models. For *seq*, the variational factor is:

$$\ln q\left(A^{(k)}\right) = \sum_{j=1}^{J}\sum_{l=1}^{J} \mathrm{Dir}\left(\left[\boldsymbol{N}_{j,l,m}^{(k)} \forall m \in \{1,..,J\}\right]\right)$$

$$N_{j,l,m}^{(k)} = \alpha_{j,l,m}^{(k)} + \sum_{n=1}^{N}\sum_{\tau=1}^{L_n} r_{n,\tau,j}\,\delta_{l,c_{n,\tau-1}^{(k)}}\,\delta_{m,c_{n,\tau}^{(k)}}, \qquad (28)$$

where δ is the Kronecker delta. For *CM*, *MACE*, *CV* and *acc*, the factors follow a similar pattern of summing pseudo-counts of correct and incorrect answers.

C Visualising Annotator Models

Figure 3 provides an alternative visualisation of the *seq* models inferred by BSC-seq for annotators in the PICO dataset. The annotators were clustered as described in Section 6 of the main paper, and the mean confusion matrices for each cluster are plotted in Figure 3 using 3D plots to emphasise the differences between the likelihoods of annotators in each cluster providing a particular label given the true label value.