Segmentation for Efficient Supervised Language Annotation with an Explicit Cost-Utility Tradeoff

Matthias Sperber[1], **Mirjam Simantzik**[2], **Graham Neubig**[3], **Satoshi Nakamura**[3], **Alex Waibel**[1]
[1]Karlsruhe Institute of Technology, Institute for Anthropomatics, Germany
[2]Mobile Technologies GmbH, Germany
[3]Nara Institute of Science and Technology, AHC Laboratory, Japan
matthias.sperber@kit.edu, mirjam.simantzik@jibbigo.com, neubig@is.naist.jp
s-nakamura@is.naist.jp, waibel@kit.edu

Abstract

In this paper, we study the problem of manually correcting automatic annotations of natural language in as efficient a manner as possible. We introduce a method for automatically segmenting a corpus into chunks such that many uncertain labels are grouped into the same chunk, while human supervision can be omitted altogether for other segments. A tradeoff must be found for segment sizes. Choosing short segments allows us to reduce the number of highly confident labels that are supervised by the annotator, which is useful because these labels are often already correct and supervising correct labels is a waste of effort. In contrast, long segments reduce the cognitive effort due to context switches. Our method helps find the segmentation that optimizes supervision efficiency by defining user models to predict the cost and utility of supervising each segment and solving a constrained optimization problem balancing these contradictory objectives. A user study demonstrates noticeable gains over pre-segmented, confidence-ordered baselines on two natural language processing tasks: speech transcription and word segmentation.

1 Introduction

Many natural language processing (NLP) tasks require human supervision to be useful in practice, be it to collect suitable training material or to meet some desired output quality. Given the high cost of human intervention, how to minimize the supervision effort is an important research problem. Previous works in areas such as active learning, post editing, and interactive pattern recognition have investigated this question with notable success (Settles, 2008; Specia, 2011; González-Rubio et al., 2010).

(a) It was a bright cold **(they)** in **(apron)**, and **(a)** clocks were striking thirteen.
(b) It was a bright cold **(they)** in **(apron)**, and **(a)** clocks were striking thirteen.
(c) It was a bright cold **(they)** in **(apron)**, and **(a)** clocks were striking thirteen.

Figure 1: Three automatic transcripts of the sentence "It was a bright cold day in April, and the clocks were striking thirteen", with recognition errors in parentheses. The underlined parts are to be corrected by a human for (a) sentences, (b) words, or (c) the proposed segmentation.

The most common framework for efficient annotation in the NLP context consists of training an NLP system on a small amount of baseline data, and then running the system on unannotated data to estimate confidence scores of the system's predictions (Settles, 2008). Sentences with the lowest confidence are then used as the data to be annotated (Figure 1 (a)). However, it has been noted that when the NLP system in question already has relatively high accuracy, annotating entire sentences can be wasteful, as most words will already be correct (Tomanek and Hahn, 2009; Neubig et al., 2011). In these cases, it is possible to achieve much higher benefit per annotated word by annotating sub-sentential units (Figure 1 (b)).

However, as Settles et al. (2008) point out, simply maximizing the benefit per annotated instance is not enough, as the real supervision effort varies

Transactions of the Association for Computational Linguistics, 2 (2014) 169–180. Action Editor: Eric Fosler-Lussier.
Submitted 11/2013; Revised 2/2014; Published 4/2014.

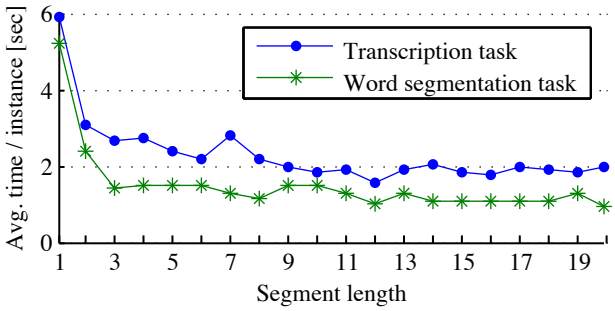

Figure 2: Average annotation time per instance, plotted over different segment lengths. For both tasks, the effort clearly increases for short segments.

greatly across instances. This is particularly important in the context of choosing segments to annotate, as human annotators heavily rely on semantics and context information to process language, and intuitively, a consecutive sequence of words can be supervised faster and more accurately than the same number of words spread out over several locations in a text. This intuition can also be seen in our empirical data in Figure 2, which shows that for the speech transcription and word segmentation tasks described later in Section 5, short segments had a longer annotation time per word. Based on this fact, we argue it would be desirable to present the annotator with a segmentation of the data into easily supervisable chunks that are both large enough to reduce the number of context switches, and small enough to prevent unnecessary annotation (Figure 1 (c)).

In this paper, we introduce a new strategy for natural language supervision tasks that attempts to optimize supervision efficiency by choosing an appropriate segmentation. It relies on a user model that, given a specific segment, predicts the cost and the utility of supervising that segment. Given this user model, the goal is to find a segmentation that minimizes the total predicted cost while maximizing the utility. We balance these two criteria by defining a constrained optimization problem in which one criterion is the optimization objective, while the other criterion is used as a constraint. Doing so allows specifying practical optimization goals such as "remove as many errors as possible given a limited time budget," or "annotate data to obtain some required classifier accuracy in as little time as possible."

Solving this optimization task is computationally difficult, an NP-hard problem. Nevertheless, we demonstrate that by making realistic assumptions about the segment length, an optimal solution can be found using an integer linear programming formulation for mid-sized corpora, as are common for supervised annotation tasks. For larger corpora, we provide simple heuristics to obtain an approximate solution in a reasonable amount of time.

Experiments over two example scenarios demonstrate the usefulness of our method: Post editing for speech transcription, and active learning for Japanese word segmentation. Our model predicts noticeable efficiency gains, which are confirmed in experiments with human annotators.

2 Problem Definition

The goal of our method is to find a segmentation over a corpus of word tokens w_1^N that optimizes supervision efficiency according to some predictive user model. The user model is denoted as a set of functions $u_{l,k}(w_a^b)$ that evaluate any possible subsequence w_a^b of tokens in the corpus according to criteria $l \in L$, and supervision modes $k \in K$.

Let us illustrate this with an example. Sperber et al. (2013) defined a framework for speech transcription in which an initial, erroneous transcript is created using automatic speech recognition (ASR), and an annotator corrects the transcript either by correcting the words by keyboard, by respeaking the content, or by leaving the words as is. In this case, we could define $K = \{\text{TYPE}, \text{RESPEAK}, \text{SKIP}\}$, each constant representing one of these three supervision modes. Our method will automatically determine the appropriate supervision mode for each segment.

The user model in this example might evaluate every segment according to two criteria L, a cost criterion (in terms of supervision time) and a utility criterion (in terms of number of removed errors), when using each mode. Intuitively, respeaking should be assigned both lower cost (because speaking is faster than typing), but also lower utility than typing on a keyboard (because respeaking recognition errors can occur). The SKIP mode denotes the special, unsupervised mode that always returns 0 cost and 0 utility.

Other possible supervision modes include multiple input modalities (Suhm et al., 2001), several human annotators with different expertise and cost

(Donmez and Carbonell, 2008), and correction vs. translation from scratch in machine translation (Specia, 2011). Similarly, cost could instead be expressed in monetary terms, or the utility function could predict the improvement of a classifier when the resulting annotation is not intended for direct human consumption, but as training data for a classifier in an active learning framework.

3 Optimization Framework

Given this setting, we are interested in simultaneously finding optimal locations and supervision modes for all segments, according to the given criteria. Each resulting segment will be assigned exactly one of these supervision modes. We denote a segmentation of the N tokens of corpus w_1^N into $M \leq N$ segments by specifying segment boundary markers $s_1^{M+1} = (s_1 = 1, s_2, \ldots, s_{M+1} = N+1)$. Setting a boundary marker $s_i = a$ means that we put a segment boundary before the a-th word token (or the end-of-corpus marker for $a = N+1$). Thus our corpus is segmented into token sequences $[(w_{s_j}, \ldots, w_{s_{j+1}-1})]_{j=1}^{M}$. The supervision modes assigned to each segment are denoted by m_j. We favor those segmentations that minimize the cumulative value $\sum_{j=1}^{M} [u_{l,m_j}(w_{s_j}^{s_{j+1}})]$ for each criterion l. For any criterion where larger values are intuitively better, we flip the sign before defining $u_{l,m_j}(w_{s_j}^{s_{j+1}})$ to maintain consistency (e.g. negative number of errors removed).

3.1 Multiple Criteria Optimization

In the case of a single criterion ($|L| = 1$), we obtain a simple, single-objective unconstrained linear optimization problem, efficiently solvable via dynamic programming (Terzi and Tsaparas, 2006). However, in practice one usually encounters several competing criteria, such as cost and utility, and here we will focus on this more realistic setting. We balance competing criteria by using one as an optimization objective, and the others as constraints.[1] Let criterion l' be the optimization objective criterion, and let C_l denote the constraining constants for the criteria $l \in L_{-l'} = L \setminus \{l'\}$. We state the optimization problem:

$$\min_{M; s_1^{M+1}; m_1^M} \sum_{j=1}^{M} \left[u_{l',m_j} \left(w_{s_j}^{s_{j+1}} \right) \right]$$

$$\text{s.t.} \sum_{j=1}^{M} \left[u_{l,m_j} \left(w_{s_j}^{s_{j+1}} \right) \right] \leq C_l \quad (\forall l \in L_{-l'})$$

This constrained optimization problem is difficult to solve. In fact, the NP-hard multiple-choice knapsack problem (Pisinger, 1994) corresponds to a special case of our problem in which the number of segments is equal to the number of tokens, implying that our more general problem is NP-hard as well.

In order to overcome this problem, we reformulate search for the optimal segmentation as a resource-constrained shortest path problem in a directed, acyclic multigraph. While still not efficiently solvable in theory, this problem is well studied in domains such as vehicle routing and crew scheduling (Irnich and Desaulniers, 2005), and it is known that in many practical situations the problem can be solved reasonably efficiently using integer linear programming relaxations (Toth and Vigo, 2001).

In our formalism, the set of nodes V represents the spaces between neighboring tokens, at which the algorithm may insert segment boundaries. A node with index i represents a segment break before the i-th token, and thus the sequence of the indices in a path directly corresponds to s_1^{M+1}. Edges E denote the grouping of tokens between the respective

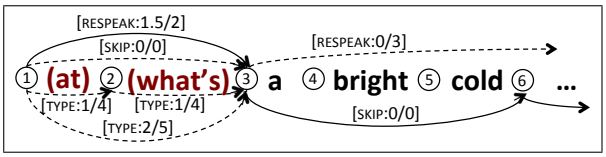

Figure 3: Excerpt of a segmentation graph for an example transcription task similar to Figure 1 (some edges are omitted for readability). Edges are labeled with their mode, predicted number of errors that can be removed, and necessary supervision time. A segmentation scheme might prefer solid edges over dashed ones in this example.

[1] This approach is known as the *bounded objective function method* in multi-objective optimization literature (Marler and Arora, 2004). The very popular *weighted sum method* merges criteria into a single efficiency measure, but is problematic in our case because the number of supervised tokens is unspecified. Unless the weights are carefully chosen, the algorithm might find, e.g., the completely unsupervised or completely supervised segmentation to be most "efficient."

nodes into one segment. Edges are always directed from left to right, and labeled with a supervision mode. In addition, each edge between nodes i and j is assigned $u_{l,k}(w_i^{j-1})$, the corresponding predicted value for each criterion $l \in L$ and supervision mode $k \in K$, indicating that the supervision mode of the j-th segment in a path directly corresponds to m_j.

Figure 3 shows an example of what the resulting graph may look like. Our original optimization problem is now equivalent to finding the shortest path between the first and last nodes according to criterion l', while obeying the given resource constraints. According to a widely used formulation for the resource constrained shortest path problem, we can define E_{ij} as the set of competing edges between i and j, and express this optimization problem with the following integer linear program (ILP):

$$\min_{\mathbf{x}} \sum_{i,j \in V} \sum_{k \in E_{ij}} x_{ijk} u_{l',k}(s_i^{j-1}) \tag{1}$$

$$\text{s.t.} \sum_{i,j \in V} \sum_{k \in E_{ij}} x_{ijk} u_{l,k}(s_i^{j-1}) \leq C_l \qquad (\forall l \in L_{-l'}) \tag{2}$$

$$\sum_{\substack{i \in V \\ k \in E_{ij}}} x_{ijk} = \sum_{\substack{i \in V \\ k \in E_{ij}}} x_{jik} \qquad (\forall j \in V \backslash \{1, n\}) \tag{3}$$

$$\sum_{\substack{j \in V \\ k \in E_{1j}}} x_{1jk} = 1 \tag{4}$$

$$\sum_{\substack{i \in V \\ k \in E_{in}}} x_{ink} = 1 \tag{5}$$

$$x_{ijk} \in \{0, 1\} \qquad (\forall x_{ijk} \in \mathbf{x}) \tag{6}$$

The variables $\mathbf{x}=\{x_{ijk} | i, j \in V, k \in E_{ij}\}$ denote the activation of the k'th edge between nodes i and j. The shortest path according to the minimization objective (1), that still meets the resource constraints for the specified criteria (2), is to be computed. The degree constraints (3,4,5) specify that all but the first and last nodes must have as many incoming as outgoing edges, while the first node must have exactly one outgoing, and the last node exactly one incoming edge. Finally, the integrality condition (6) forces all edges to be either fully activated or fully deactivated. The outlined problem formulation can solved directly by using off-the-shelf ILP solvers, here we employ GUROBI (Gurobi Optimization, 2012).

3.2 Heuristics for Approximation

In general, edges are inserted for every supervision mode between every combination of two nodes. The search space can be constrained by removing some of these edges to increase efficiency. In this study, we only consider edges spanning at most 20 tokens.

For cases in which larger corpora are to be annotated, or when the acceptable delay for delivering results is small, a suitable segmentation can be found approximately. The easiest way would be to partition the corpus, e.g. according to its individual documents, divide the budget constraints evenly across all partitions, and then segment each partition independently. More sophisticated methods might approximate the Pareto front for each partition, and distribute the budgets in an intelligent way.

4 User Modeling

While the proposed framework is able to optimize the segmentation with respect to each criterion, it also rests upon the assumption that we can provide user models $u_{l,k}(w_i^{j-1})$ that accurately evaluate every segment according to the specified criteria and supervision modes. In this section, we discuss our strategies for estimating three conceivable criteria: annotation cost, correction of errors, and improvement of a classifier.

4.1 Annotation Cost Modeling

Modeling cost requires solving a regression problem from features of a candidate segment to annotation cost, for example in terms of supervision time. Appropriate input features depend on the task, but should include notions of complexity (e.g. a confidence measure) and length of the segment, as both are expected to strongly influence supervision time.

We propose using Gaussian process (GP) regression for cost prediction, a start-of-the-art nonparametric Bayesian regression technique (Rasmussen and Williams, 2006)[2]. As reported on a similar task by Cohn and Specia (2013), and confirmed by our preliminary experiments, GP regression significantly outperforms popular techniques such as sup-

[2]Code available at http://www.gaussianprocess.org/gpml/

port vector regression and least-squares linear regression. We also follow their settings for GP, employing GP regression with a squared exponential kernel with automatic relevance determination. Depending on the number of users and amount of training data available for each user, models may be trained separately for each user (as we do here), or in a combined fashion via multi-task learning as proposed by Cohn and Specia (2013).

It is also crucial for the predictions to be reliable throughout the whole relevant space of segments. If the cost of certain types of segments is systematically underpredicted, the segmentation algorithm might be misled to prefer these, possibly a large number of times.[3] An effective trick to prevent such underpredictions is to predict the log time instead of the actual time. In this way, errors in the critical low end are penalized more strongly, and the time can never become negative.

4.2 Error Correction Modeling

As one utility measure, we can use the number of errors corrected, a useful measure for post editing tasks over automatically produced annotations. In order to measure how many errors can be removed by supervising a particular segment, we must estimate both how many errors are in the automatic annotation, and how reliably a human can remove these for a given supervision mode.

Most machine learning techniques can estimate confidence scores in the form of posterior probabilities. To estimate the number of errors, we can sum over one minus the posterior for all tokens, which estimates the Hamming distance from the reference annotation. This measure is appropriate for tasks in which the number of tokens is fixed in advance (e.g. a part-of-speech estimation task), and a reasonable approximation for tasks in which the number of tokens is not known in advance (e.g. speech transcription, cf. Section 5.1.1).

Predicting the particular tokens at which a human will make a mistake is known to be a difficult task (Olson and Olson, 1990), but a simplifying constant human error rate can still be useful. For example, in the task from Section 2, we may suspect a certain number of errors in a transcript segment, and predict, say, 95% of those errors to be removed via typing, but only 85% via respeaking.

4.3 Classifier Improvement Modeling

Another reasonable utility measure is accuracy of a classifier trained on the data we choose to annotate in an active learning framework. Confidence scores have been found useful for ranking particular tokens with regards to how much they will improve a classifier (Settles, 2008). Here, we may similarly score segment utility as the sum of its token confidences, although care must be taken to normalize and calibrate the token confidences to be linearly comparable before doing so. While the resulting utility score has no interpretation in absolute terms, it can still be used as an optimization objective (cf. Section 5.2.1).

5 Experiments

In this section, we present experimental results examining the effectiveness of the proposed method over two tasks: speech transcription and Japanese word segmentation.[4]

5.1 Speech Transcription Experiments

Accurate speech transcripts are a much-demanded NLP product, useful by themselves, as training material for ASR, or as input for follow-up tasks like speech translation. With recognition accuracies plateauing, manually correcting (post editing) automatic speech transcripts has become popular. Common approaches are to identify words (Sanchez-Cortina et al., 2012) or (sub-)sentences (Sperber et al., 2013) of low confidence, and have a human editor correct these.

5.1.1 Experimental Setup

We conduct a user study in which participants post-edited speech transcripts, given a fixed goal word error rate. The transcription setup was such that the transcriber could see the ASR transcript of parts before and after the segment that he was editing, providing context if needed. When imprecise time alignment resulted in segment breaks that were

[3] For instance, consider a model that predicts well for segments of medium size or longer, but underpredicts the supervision time of single-token segments. This may lead the segmentation algorithm to put every token into its own segment, which is clearly undesirable.

[4] Software and experimental data can be downloaded from http://www.msperber.com/research/tacl-segmentation/

slightly "off," as happened occasionally, that context helped guess what was said. The segment itself was transcribed from scratch, as opposed to editing the ASR transcript; besides being arguably more efficient when the ASR transcript contains many mistakes (Nanjo et al., 2006; Akita et al., 2009), preliminary experiments also showed that supervision time is far easier to predict this way. Figure 4 illustrates what the setup looked like.

We used a self-developed transcription tool to conduct experiments. It presents our computed segments one by one, allows convenient input and playback via keyboard shortcuts, and logs user interactions with their time stamps. A selection of TED talks[5] (English talks on technology, entertainment, and design) served as experimental data. While some of these talks contain jargon such as medical terms, they are presented by skilled speakers, making them comparably easy to understand. Initial transcripts were created using the Janus recognition toolkit (Soltau et al., 2001) with a standard, TED-optimized setup. We used confusion networks for decoding and obtaining confidence scores.

For reasons of simplicity, and better comparability to our baseline, we restricted our experiment to two supervision modes: TYPE and SKIP. We conducted experiments with 3 participants, 1 with several years of experience in transcription, 2 with none. Each participant received an explanation on the transcription guidelines, and a short hands-on training to learn to use our tool. Next, they transcribed a balanced selection of 200 segments of varying length and quality in random order. This data was used to train the user models.

Finally, each participant transcribed another 2 TED talks, with word error rate (WER) 19.96% (predicted: 22.33%). We set a target (predicted) WER of 15% as our optimization constraint,[6] and minimize the predicted supervision time as our objective function. Both TED talks were transcribed once using the baseline strategy, and once using the proposed strategy. The order of both strategies was reversed between talks, to minimize learning bias due to transcribing each talk twice.

The baseline strategy was adopted according to Sperber et al. (2013): We segmented the talk into natural, subsentential units, using Matusov et al. (2006)'s segmenter, which we tuned to reproduce the TED subtitle segmentation, producing a mean segment length of 8.6 words. Segments were added in order of increasing average word confidence, until the user model predicted a WER$<15\%$. The second segmentation strategy was the proposed method, similarly with a resource constraint of WER$<15\%$.

Supervision time was predicted via GP regression (cf. Section 4.1), using segment length, audio duration, and mean confidence as input features. The output variable was assumed subject to additive Gaussian noise with zero mean, a variance of 5 seconds was chosen empirically to minimize the mean squared error. Utility prediction (cf. Section 4.2) was based on posterior scores obtained from the confusion networks. We found it important to calibrate them, as the posteriors were overconfident especially in the upper range. To do so, we automatically transcribed a development set of TED data, grouped the recognized words into buckets according to their posteriors, and determined the average number of errors per word in each bucket from an alignment with the reference transcript. The mapping from average posterior to average number of errors was estimated via GP regression. The result was summed over all tokens, and multiplied by a constant human confidence, separately determined for each participant.[7]

5.1.2 Simulation Results

To convey a better understanding of the potential gains afforded by our method, we first present a simulated experiment. We assume a transcriber who makes no mistakes, and needs exactly the amount of time predicted by a user model trained on the data of a randomly selected participant. We compare three scenarios: A baseline simulation, in which the baseline segments are transcribed in ascending order of confidence; a simulation using the proposed method, in which we change the WER constraint in small increments; finally, an oracle simulation, which uses

[5] www.ted.com

[6] Depending on the level of accuracy required by our final application, this target may be set lower or higher.

[7] More elaborate methods for WER estimation exist, such as by Ogawa et al. (2013), but if our method achieves improvements using simple Hamming distance, incorporating more sophisticated measures will likely achieve similar, or even better accuracy.

(3) SKIP: “nineteen forty six until today you see the green”
(4) TYPE: <annotator types: “is the traditional”>
(5) SKIP: “Interstate conflict”
(6) TYPE: <annotator types: “the ones we used to”>
(7) SKIP: . . .

Figure 4: Result of our segmentation method (excerpt). TYPE segments are displayed empty and should be transcribed from scratch. For SKIP segments, the ASR transcript is displayed to provide context. When annotating a segment, the corresponding audio is played back.

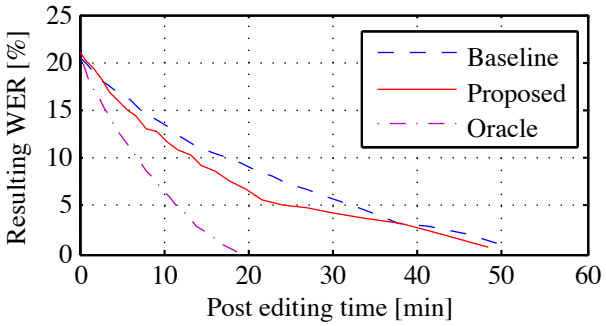

Figure 5: Simulation of post editing on example TED talk. The proposed method reduces the WER considerably faster than the baseline at first, later both converge. The much superior oracle simulation indicates room for further improvement.

the proposed method, but uses a utility model that knows the actual number of errors in each segment. For each supervised segment, we simply replace the ASR output with the reference, and measure the resulting WER.

Figure 5 shows the simulation on an example TED talk, based on an initial transcript with 21.9% WER. The proposed method is able to reduce the WER faster than the baseline, up to a certain point where they converge. The oracle simulation is even faster, indicating room for improvement through better confidence scores.

5.1.3 User Study Results

Table 1 shows the results of the user study. First, we note that the WER estimation by our utility model was off by about 2.5%: While the predicted improvement in WER was from 22.33% to 15.0%, the actual improvement was from 19.96% to about 12.5%. The actual resulting WER was consistent

Participant	Baseline		Proposed	
	WER	Time	WER	Time
P_1	12.26	44:05	12.18	33:01
P_2	12.75	36:19	12.77	29:54
P_3	12.70	52:42	12.50	37:57
AVG	12.57	44:22	12.48	33:37

Table 1: Transcription task results. For each user, the resulting WER [%] after supervision is shown, along with the time [min] they needed. The unsupervised WER was 19.96%.

across all users, and we observe strong, consistent reductions in supervision time for all participants. Prediction of the necessary supervision time was accurate: Averaged over participants, 45:41 minutes were predicted for the baseline, 44:22 minutes measured. For the proposed method, 32:11 minutes were predicted, 33:37 minutes measured. On average, participants removed 6.68 errors per minute using the baseline, and 8.93 errors per minute using the proposed method, a speed-up of 25.2%.

Note that predicted and measured values are not strictly comparable: In the experiments, to provide a fair comparison participants transcribed the same talks twice (once using baseline, once the proposed method, in alternating order), resulting in a noticeable learning effect. The user model, on the other hand, is trained to predict the case in which a transcriber conducts only one transcription pass.

As an interesting finding, without being informed about the order of baseline and proposed method, participants reported that transcribing according to the proposed segmentation seemed harder, as they found the baseline segmentation more linguistically reasonable. However, this perceived increase in difficulty did not show in efficiency numbers.

5.2 Japanese Word Segmentation Experiments

Word segmentation is the first step in NLP for languages that are commonly written without word boundaries, such as Japanese and Chinese. We apply our method to a task in which we domain-adapt a word segmentation classifier via active learning. In this experiment, participants annotated whether or not a word boundary occurred at certain positions in a Japanese sentence. The tokens to be grouped into segments are positions between adjacent characters.

5.2.1 Experimental Setup

Neubig et al. (2011) have proposed a pointwise method for Japanese word segmentation that can be trained using partially annotated sentences, which makes it attractive in combination with active learning, as well as our segmentation method. The authors released their method as a software package "KyTea" that we employed in this user study. We used KyTea's active learning domain adaptation toolkit[8] as a baseline.

For data, we used the Balanced Corpus of Contemporary Written Japanese (BCCWJ), created by Maekawa (2008), with the *internet Q&A subcorpus* as in-domain data, and the *whitepaper* subcorpus as background data, a domain adaptation scenario. Sentences were drawn from the in-domain corpus, and the manually annotated data was then used to train KyTea, along with the pre-annotated background data. The goal (objective function) was to improve KyTea's classification accuracy on an in-domain test set, given a constrained time budget of 30 minutes. There were again 2 supervision modes: ANNOTATE and SKIP. Note that this is essentially a batch active learning setup with only one iteration.

We conducted experiments with one expert with several years of experience with Japanese word segmentation annotation, and three non-expert native speakers with no prior experience. Japanese word segmentation is not a trivial task, so we provided non-experts with training, including explanation of the segmentation standard, a supervised test with immediate feedback and explanations, and hands-on training to get used to the annotation software.

Supervision time was predicted via GP regression (cf. Section 4.1), using the segment length and mean confidence as input features. As before, the output variable was assumed subject to additive Gaussian noise with zero mean and 5 seconds variance. To obtain training data for these models, each participant annotated about 500 example instances, drawn from the adaptation corpus, grouped into segments and balanced regarding segment length and difficulty.

For utility modeling (cf. Section 4.3), we first normalized KyTea's confidence scores, which are given in terms of SVM margin, using a sigmoid function (Platt, 1999). The normalization parameter was selected so that the mean confidence on a development set corresponded to the actual classifier accuracy. We derive our measure of classifier improvement for correcting a segment by summing over one minus the calibrated confidence for each of its tokens. To analyze how well this measure describes the actual training utility, we trained KyTea using the background data plus disjoint groups of 100 in-domain instances with similar probabilities and measured the achieved reduction of prediction errors. The correlation between each group's mean utility and the achieved error reduction was 0.87. Note that we ignore the decaying returns usually observed as more data is added to the training set. Also, we did not attempt to model user errors. Employing a constant base error rate, as in the transcription scenario, would change segment utilities only by a constant factor, without changing the resulting segmentation.

After creating the user models, we conducted the main experiment, in which each participant annotated data that was selected from a pool of 1000 in-domain sentences using two strategies. The first, baseline strategy was as proposed by Neubig et al. (2011). Queries are those instances with the lowest confidence scores. Each query is then extended to the left and right, until a word boundary is predicted. This strategy follows similar reasoning as was the premise to this paper: To decide whether or not a position in a text corresponds to a word boundary, the annotator has to acquire surrounding context information. This context acquisition is relatively time consuming, so he might as well label the surrounding instances with little additional effort. The second strategy was our proposed, more principled approach. Queries of both methods were shuffled to minimize bias due to learning effects. Finally, we trained KyTea using the results of both methods, and compared the achieved classifier improvement and supervision times.

5.2.2 User Study Results

Table 2 summarizes the results of our experiment. It shows that the annotations by each participant resulted in a better classifier for the proposed method than the baseline, but also took up considerably more time, a less clear improvement than for the transcription task. In fact, the total error for time predictions was as high as 12.5% on average,

[8] http://www.phontron.com/kytea/active.html

Participant	Baseline		Proposed	
	Time	Acc.	Time	Acc.
Expert	25:50	96.17	32:45	96.55
NonExp_1	22:05	95.79	26:44	95.98
NonExp_2	23:37	96.15	31:28	96.21
NonExp_3	25:23	96.38	33:36	96.45

Table 2: Word segmentation task results, for our expert and 3 non-expert participants. For each participant, the resulting classifier accuracy [%] after supervision is shown, along with the time [min] they needed. The unsupervised accuracy was 95.14%.

where the baseline method tended take less time than predicted, the proposed method more time. This is in contrast to a much lower total error (within 1%) when cross-validating our user model training data. This is likely due to the fact that the data for training the user model was selected in a balanced manner, as opposed to selecting difficult examples, as our method is prone to do. Thus, we may expect much better predictions when selecting user model training data that is more similar to the test case.

Plotting classifier accuracy over annotation time draws a clearer picture. Let us first analyze the results for the expert annotator. Figure 6 (E.1) shows that the proposed method resulted in consistently better results, indicating that time predictions were still effective. Note that this comparison may put the proposed method at a slight disadvantage by comparing intermediate results despite optimizing globally.

For the non-experts, the improvement over the baseline is less consistent, as can be seen in Figure 6 (N.1) for one representative. According to our analysis, this can be explained by two factors: (1) The non-experts' annotation error (6.5% on average) was much higher than the expert's (2.7%), resulting in a somewhat irregular classifier learning curve. (2) The variance in annotation time per segment was consistently higher for the non-experts than the expert, indicated by an average per-segment prediction error of 71% vs. 58% relative to the mean actual value, respectively. Informally speaking, non-experts made more mistakes, and were more strongly influenced by the difficulty of a particular segment (which was higher on average with the proposed method, as indicated by a lower average confidence).[9]

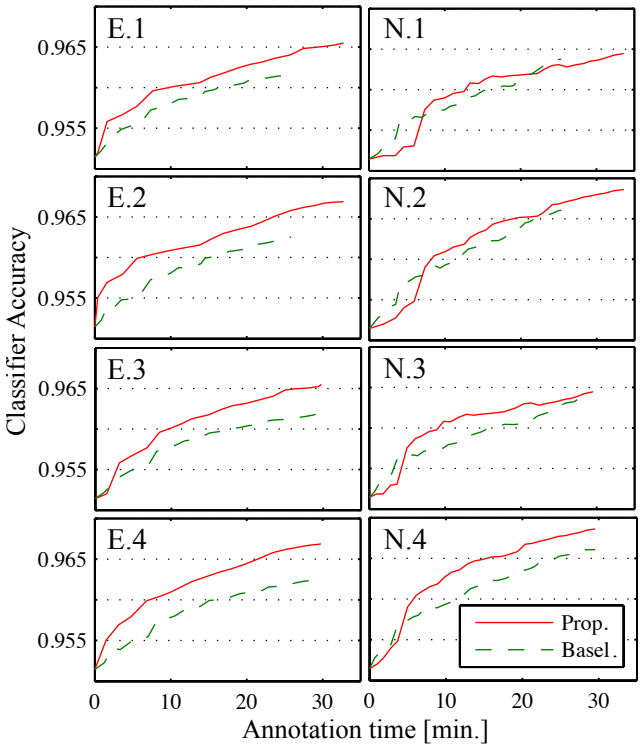

Figure 6: Classifier improvement over time, depicted for the expert (E) and a non-expert (N). The graphs show numbers based on (1) actual annotations and user models as in Sections 4.1 and 4.3, (2) error-free annotations, (3) measured times replaced by predicted times, and (4) both reference annotations and replaced time predictions.

In Figures 6 (2-4) we present a simulation experiment in which we first pretend as if annotators made no mistakes, then as if they needed exactly as much time as predicted for each segment, and then both. This cheating experiment works in favor of the proposed method, especially for the non-expert. We may conclude that our segmentation approach is effective for the word segmentation task, but requires more accurate time predictions. Better user models will certainly help, although for the presented scenario our method may be most useful for an expert annotator.

[9] Note that the non-expert in the figure annotated much faster than the expert, which explains the comparable classification result despite making more annotation errors. This is in contrast to the other non-experts, who were slower.

5.3 Computational Efficiency

Since our segmentation algorithm does not guarantee polynomial runtime, computational efficiency was a concern, but did not turn out problematic. On a consumer laptop, the solver produced segmentations within a few seconds for a single document containing several thousand tokens, and within hours for corpora consisting of several dozen documents. Runtime increased roughly quadratically with respect to the number of segmented tokens. We feel that this is acceptable, considering that the time needed for human supervision will likely dominate the computation time, and reasonable approximations can be made as noted in Section 3.2.

6 Relation to Prior Work

Efficient supervision strategies have been studied across a variety of NLP-related research areas, and received increasing attention in recent years. Examples include post editing for speech recognition (Sanchez-Cortina et al., 2012), interactive machine translation (González-Rubio et al., 2010), active learning for machine translation (Haffari et al., 2009; González-Rubio et al., 2011) and many other NLP tasks (Olsson, 2009), to name but a few studies.

It has also been recognized by the active learning community that correcting the most useful parts first is often not optimal in terms of efficiency, since these parts tend to be the most difficult to manually annotate (Settles et al., 2008). The authors advocate the use of a user model to predict the supervision effort, and select the instances with best "bang-for-the-buck." This prediction of supervision effort was successful, and was further refined in other NLP-related studies (Tomanek et al., 2010; Specia, 2011; Cohn and Specia, 2013). Our approach to user modeling using GP regression is inspired by the latter.

Most studies on user models consider only supervision effort, while neglecting the accuracy of human annotations. The view on humans as a perfect oracle has been criticized (Donmez and Carbonell, 2008), since human errors are common and can negatively affect supervision utility. Research on human-computer-interaction has identified the modeling of human errors as very difficult (Olson and Olson, 1990), depending on factors such as user experience, cognitive load, user interface design, and fatigue. Nevertheless, even the simple error model used in our post editing task was effective.

The active learning community has addressed the problem of balancing utility and cost in some more detail. The previously reported "bang-for-the-buck" approach is a very simple, greedy approach to combine both into one measure. A more theoretically founded scalar optimization objective is the net benefit (utility minus costs) as proposed by Vijayanarasimhan and Grauman (2009), but unfortunately is restricted to applications where both can be expressed in terms of the same monetary unit. Vijayanarasimhan et al. (2010) and Donmez and Carbonell (2008) use a more practical approach that specifies a constrained optimization problem by allowing only a limited time budget for supervision. Our approach is a generalization thereof and allows either specifying an upper bound on the predicted cost, or a lower bound on the predicted utility.

The main novelty of our presented approach is the explicit modeling and selection of segments of various sizes, such that annotation efficiency is optimized according to the specified constraints. While some works (Sassano and Kurohashi, 2010; Neubig et al., 2011) have proposed using subsentential segments, we are not aware of any previous work that explicitly optimizes that segmentation.

7 Conclusion

We presented a method that can effectively choose a segmentation of a language corpus that optimizes supervision efficiency, considering not only the actual usefulness of each segment, but also the annotation cost. We reported noticeable improvements over strong baselines in two user studies. Future user experiments with more participants would be desirable to verify our observations, and allow further analysis of different factors such as annotator expertise. Also, future research may improve the user modeling, which will be beneficial for our method.

Acknowledgments

The research leading to these results has received funding from the European Union Seventh Framework Programme (FP7/2007-2013) under grant agreement n 287658 Bridges Across the Language Divide (EU-BRIDGE).

References

Yuya Akita, Masato Mimura, and Tatsuya Kawahara. 2009. Automatic Transcription System for Meetings of the Japanese National Congress. In *Interspeech*, pages 84–87, Brighton, UK.

Trevor Cohn and Lucia Specia. 2013. Modelling Annotator Bias with Multi-task Gaussian Processes: An Application to Machine Translation Quality Estimation. In *Association for Computational Linguistics Conference (ACL)*, Sofia, Bulgaria.

Pinar Donmez and Jaime Carbonell. 2008. Proactive Learning : Cost-Sensitive Active Learning with Multiple Imperfect Oracles. In *Conference on Information and Knowledge Management (CIKM)*, pages 619–628, Napa Valley, CA, USA.

Jesús González-Rubio, Daniel Ortiz-Martínez, and Francisco Casacuberta. 2010. Balancing User Effort and Translation Error in Interactive Machine Translation Via Confidence Measures. In *Association for Computational Linguistics Conference (ACL), Short Papers Track*, pages 173–177, Uppsala, Sweden.

Jesús González-Rubio, Daniel Ortiz-Martínez, and Francisco Casacuberta. 2011. An active learning scenario for interactive machine translation. In *International Conference on Multimodal Interfaces (ICMI)*, pages 197–200, Alicante, Spain.

Gurobi Optimization. 2012. Gurobi Optimizer Reference Manual.

Gholamreza Haffari, Maxim Roy, and Anoop Sarkar. 2009. Active Learning for Statistical Phrase-based Machine Translation. In *North American Chapter of the Association for Computational Linguistics - Human Language Technologies Conference (NAACL-HLT)*, pages 415–423, Boulder, CO, USA.

Stefan Irnich and Guy Desaulniers. 2005. Shortest Path Problems with Resource Constraints. In *Column Generation*, pages 33–65. Springer US.

Kikuo Maekawa. 2008. Balanced Corpus of Contemporary Written Japanese. In *International Joint Conference on Natural Language Processing (IJCNLP)*, pages 101–102, Hyderabad, India.

R. Timothy Marler and Jasbir S. Arora. 2004. Survey of multi-objective optimization methods for engineering. *Structural and Multidisciplinary Optimization*, 26(6):369–395, April.

Evgeny Matusov, Arne Mauser, and Hermann Ney. 2006. Automatic Sentence Segmentation and Punctuation Prediction for Spoken Language Translation. In *International Workshop on Spoken Language Translation (IWSLT)*, pages 158–165, Kyoto, Japan.

Hiroaki Nanjo, Yuya Akita, and Tatsuya Kawahara. 2006. Computer Assisted Speech Transcription System for Efficient Speech Archive. In *Western Pacific Acoustics Conference (WESPAC)*, Seoul, Korea.

Graham Neubig, Yosuke Nakata, and Shinsuke Mori. 2011. Pointwise Prediction for Robust , Adaptable Japanese Morphological Analysis. In *Association for Computational Linguistics: Human Language Technologies Conference (ACL-HLT)*, pages 529–533, Portland, OR, USA.

Atsunori Ogawa, Takaaki Hori, and Atsushi Nakamura. 2013. Discriminative Recognition Rate Estimation For N-Best List and Its Application To N-Best Rescoring. In *International Conference on Acoustics, Speech, and Signal Processing (ICASSP)*, pages 6832–6836, Vancouver, Canada.

Judith Reitman Olson and Gary Olson. 1990. The Growth of Cognitive Modeling in Human-Computer Interaction Since GOMS. *Human-Computer Interaction*, 5(2):221–265, June.

Fredrik Olsson. 2009. A literature survey of active machine learning in the context of natural language processing. Technical report, SICS Sweden.

David Pisinger. 1994. A Minimal Algorithm for the Multiple-Choice Knapsack Problem. *European Journal of Operational Research*, 83(2):394–410.

John C. Platt. 1999. Probabilistic Outputs for Support Vector Machines and Comparisons to Regularized Likelihood Methods. In *Advances in Large Margin Classifiers*, pages 61–74. MIT Press.

Carl E. Rasmussen and Christopher K.I. Williams. 2006. *Gaussian Processes for Machine Learning*. MIT Press, Cambridge, MA, USA.

Isaias Sanchez-Cortina, Nicolas Serrano, Alberto Sanchis, and Alfons Juan. 2012. A prototype for Interactive Speech Transcription Balancing Error and Supervision Effort. In *International Conference on Intelligent User Interfaces (IUI)*, pages 325–326, Lisbon, Portugal.

Manabu Sassano and Sadao Kurohashi. 2010. Using Smaller Constituents Rather Than Sentences in Active Learning for Japanese Dependency Parsing. In *Association for Computational Linguistics Conference (ACL)*, pages 356–365, Uppsala, Sweden.

Burr Settles, Mark Craven, and Lewis Friedland. 2008. Active Learning with Real Annotation Costs. In *Neural Information Processing Systems Conference (NIPS) - Workshop on Cost-Sensitive Learning*, Lake Tahoe, NV, United States.

Burr Settles. 2008. An Analysis of Active Learning Strategies for Sequence Labeling Tasks. In *Conference on Empirical Methods in Natural Language Processing (EMNLP)*, pages 1070–1079, Honolulu, USA.

Hagen Soltau, Florian Metze, Christian Fügen, and Alex Waibel. 2001. A One-Pass Decoder Based on Polymorphic Linguistic Context Assignment. In *Automatic Speech Recognition and Understanding Work-*

shop (ASRU), pages 214–217, Madonna di Campiglio, Italy.

Lucia Specia. 2011. Exploiting Objective Annotations for Measuring Translation Post-editing Effort. In *Conference of the European Association for Machine Translation (EAMT)*, pages 73–80, Nice, France.

Matthias Sperber, Graham Neubig, Christian Fügen, Satoshi Nakamura, and Alex Waibel. 2013. Efficient Speech Transcription Through Respeaking. In *Interspeech*, pages 1087–1091, Lyon, France.

Bernhard Suhm, Brad Myers, and Alex Waibel. 2001. Multimodal error correction for speech user interfaces. *Transactions on Computer-Human Interaction*, 8(1):60–98.

Evimaria Terzi and Panayiotis Tsaparas. 2006. Efficient algorithms for sequence segmentation. In *SIAM Conference on Data Mining (SDM)*, Bethesda, MD, USA.

Katrin Tomanek and Udo Hahn. 2009. Semi-Supervised Active Learning for Sequence Labeling. In *International Joint Conference on Natural Language Processing (IJCNLP)*, pages 1039–1047, Singapore.

Katrin Tomanek, Udo Hahn, and Steffen Lohmann. 2010. A Cognitive Cost Model of Annotations Based on Eye-Tracking Data. In *Association for Computational Linguistics Conference (ACL)*, pages 1158–1167, Uppsala, Sweden.

Paolo Toth and Daniele Vigo. 2001. *The Vehicle Routing Problem*. Society for Industrial & Applied Mathematics (SIAM), Philadelphia.

Sudheendra Vijayanarasimhan and Kristen Grauman. 2009. Whats It Going to Cost You?: Predicting Effort vs. Informativeness for Multi-Label Image Annotations. In *Conference on Computer Vision and Pattern Recognition (CVPR)*, pages 2262–2269, Miami Beach, FL, USA.

Sudheendra Vijayanarasimhan, Prateek Jain, and Kristen Grauman. 2010. Far-sighted active learning on a budget for image and video recognition. In *Conference on Computer Vision and Pattern Recognition (CVPR)*, pages 3035–3042, San Francisco, CA, USA, June.