transLectures

Transcription and Translation of Video Lectures

D4.1.2: Second report on intelligent interaction

UPVLC, XEROX, JSI-K4A, RWTH and EML

Distribution: Public

transLectures
Transcription and Translation of Video Lectures

ICT Project 287755 Deliverable D4.1.2

November 8, 2013

SEVENTH FRAMEWORK PROGRAMME

Project funded by the European Community under the Seventh Framework Programme for Research and Technological Development.

Information Society Technologies

Project ref no.	ICT-287755
Project acronym	transLectures
Project full title	Transcription and Translation of Video Lectures
Instrument	STREP
Thematic Priority	ICT-2011.4.2 Language Technologies
Start date / duration	01 November 2011 / 36 Months

Distribution	Public
Contractual date of delivery	October 31, 2013
Actual date of delivery	November 8, 2013
Date of last update	November 8, 2013
Deliverable number	D4.1.2
Deliverable title	Second report on intelligent interaction
Type	Report
Status & version	v1.0
Number of pages	40
Contributing WP(s)	WP4
WP / Task responsible	XEROX
Other contributors	
Internal reviewer	Jorge Civera, Alfons Juan
Author(s)	UPVLC, XEROX, JSI-K4A, RWTH and EML
EC project officer	Susan Fraser

The partners in transLectures are:

Universitat Politècnica de València (UPVLC)
XEROX Research Center Europe (XEROX)
Josef Stefan Institute (JSI) and its third party Knowledge for All Foundation (K4A)
RWTH Aachen University (RWTH)
European Media Laboratory GmbH (EML)
Deluxe Digital Studios Limited (DDS)

For copies of reports, updates on project activities and other transLectures related information, contact:

The transLectures Project Co-ordinator
Alfons Juan, Universitat Politècnica de València
Camí de Vera s/n, 46018 València, Spain
ajuan@dsic.upv.es
Phone +34 699-307-095 - Fax +34 963-877-359

Copies of reports and other material can also be accessed via the project's homepage: http://www.translectures.eu

Executive Summary

Contents

1 Introduction

During M13-M24, the progress in the WP4 about **Intelligent Interaction** can be summarized as follows.

Concerning the **Constrained Search** (CS) task, in the case of ASR (i.e. transcription), we extended the approaches developed during the first year in the following directions. The first approach based on constraining individual words of the transcribed sentence was extended toward constraining phrases. The second approach based on constraining a prefix of the transcription was extended towards constraining infixes by breaking down the search into multiple prefix-constrained searches. Experiments were conducted to evaluate these extensions in both approaches. In the case of the first approach, evaluated on the poliMedia corpus, the improvements already observed when using CS for single words were not significantly strengthened by using phrases. In the case of the prefix-based approach, which was applied for the first time to the VideoLectures.NET corpus, improvements from CS were observed with several different selection criteria for directing user supervision.

Constrained search was also applied to the case of translation. In this context, on top of options for constraining the translation to employ certain specific words already developed in the first year, an option was added to allow the user to specify certain source-target phrase pairs in order to guide the translation process. This option was evaluated in particular for the case of unknown source words, and was shown to lead to significant improvements in terms of TER and (to a lesser extent) BLEU score.

Relative to the **Intelligent Interaction for Transcription** task, progress was made on the following aspects. First, a new experimental setup was developed in order to better fit the transLectures application. In this setup, the videos to be transcribed are divided into several consecutive blocks; the first block is automatically transcribed using ASR, is partially supervised, and then the resulting partially corrected transcriptions are used to adapt the ASR system; the process is then repeated on the second block, using the adapted system obtained from the first block, and so on for the remaining blocks. Two types of scenarios are compared, a Batch Interaction scenario (BI) in which the available supervision resources are simply applied sequentially to the transcribed sentences according to their order in the videos, and an Intelligent Interaction scenario, where these resources are applied according to a confidence measure which attempts to select the transcriptions most requiring supervision. Experiments show that II significantly outperforms BI. This is due in part to a new, more efficient Confidence Estimation technique based on a Naive Bayes classifier. Experiments are conducted on several dimensions: the type of supervision unit chosen (words, phrases, or sentences), the type of constrained search employed (word-based vs. phrase-based), as well as a comparison between different versions of confidence estimation.

Moving now to the **Intelligent Interaction for Translation** task, a new experimental setup was also developed during the period, broadly similar to the one just described in the Transcription context. As in the transcription case, Batch Interaction was experimentally compared to Intelligent Interaction (similar definitions to the transcription case), however with less significant improvements so far in the context of translation. We conducted a deep analysis of II when sentence-level interaction units were chosen, and we extended II to use individual words as interaction units. We also experimented with several confidence measures at the word level. While the results of II are not yet as encouraging as those for transcription, which may be due in part to the fact that the user supervisions were simulated on the basis of reference translations and not actual human actions, the conducted experiments point to the value of supervision units smaller than a sentence, and we plan to pursue this direction in the next period.

Finally, concerning the **Incremental Training for Translation** task, we focused during the M13-M24 period on an approach consisting in"quick-update" short-cycles updates based on "mini-batches" of new supervised translations. We performed alignment and phrase-extraction from the mini-batches and combined the resulting phrase-tables with the tables obtained by slower long-cycles updates based on batch-training. Different configurations were evaluated and were shown to be superior to an incremental training model based on Incremental-Giza and on suffix arrays. These techniques make it possible to quickly (e.g. on a daily basis) incorporate newly supervised translations into the transLectures training workflow while performing a full retraining of the models with a slower periodicity, perhaps once a week.

2 Fast Constrained Search

2.1 Constrained Search for Transcription

2.1.1 UPVLC

In D4.1.1 UPVLC proposed a generalized ASR approach to user constraints. Specifically, this proposal was based on a modification of the usual decoding formulation in ASR to allow user constraints. Constraints correspond to speech segments which have been supervised by users. Thus, these constraints are considered as conditions that have to be met in the recomputation of hypotheses.

Specifically, a constraint is defined as $c = (w_1^m, b, e, i)$, where (b,e) are the time boundaries of the constraint, w_1^m are the words supervised by the user, and $i = \{0, 1\}$ indicates whether the words w_1^m must appear ($i = 1$ or not $i = 0$) within the segment from b to e. Constraints with $i = 1$ correspond to a supervised segment in which the user has verified that some words appear in the transcription. Meanwhile, $i = 0$ constraints correspond to the case in which the user cannot annotate the words of the given segment, and he only indicates that the recognised words are incorrect. Summarizing, given a set of constraints $C = c_1, \cdots, c_M$ and an speech segment x_1^T, the most probable transcription $\hat{w}_1^N$ is obtained as

$$x_1^T \rightarrow \hat{w}_1^N(x_1^T) \quad = \quad \underset{w_1^N}{\text{argmax}} \left\{ p(w_1^N) p(x_1^T | w_1^N, \mathcal{C}) \right\} \tag{1}$$

which is simplified to

$$\hat{w}_1^N(x_1^T) \approx \underset{w_1^N}{\text{argmax}} \left\{ \left[\prod_{n=1}^{N} p(w_n | w_1^{n-1}) \right] \cdot \underset{s_1^T}{\max} \left\{ \prod_{t=1}^{T} p(x_t | s_t, w_1^N) \cdot p(s_t | s_1^{t-1}, w_1^N, \mathcal{C}) \right\} \right\} \tag{2}$$

where $p(s_t | s_1^{t-1}, w_1^N, \mathcal{C})$ is given by

$$p(s_t | s_1^{t-1}, w_1^N, \mathcal{C}) = \begin{cases} \frac{p(s_t | s_{t-1}, w_1^N)}{\sum_{s' \in \mathcal{S}_t} p(s' | s_{t-1}, w_1^N)} & \text{if } s_\beta^t \text{ is compatible with } \mathcal{C}_t \\ 0 & \text{otherwise} \end{cases} \tag{3}$$

as observed, in this equation, only the acoustic modeling is affected by the constraint.

During the first year of the project, Eq. 1 was implemented in a simplified form by considering that sequence of words in constraints are not allowed. This means that user constraints could be defined as $c = (w, b, e, i)$. During M13-M24, Eq. 1 has been implemented in full. This means that two different implementations of constrained search are available at M24: word-based or phrase-based constrained search. The main difference between the two approaches is that constrained search at phrase-level is less restrictive than constrained search at word-level. This is because word-level constraints include time boundaries for each decoding word whereas phrase-level constraints include time boundaries only for the first and the last word of the segment.

Both approaches have been empirically compared within the scenario of intelligent interaction for transcription, but minor differences were found between word and phrase level constraints in this scenario (details in Section 3). Also, phrase-based constrained search has been added as a new feature to the TLK toolkit (see D3.2.2 for more details about TLK).

Table 1: Transcription quality in terms of WER calculated on the VideoLectures.NET eval-set using **single pass** recognition. The columns "Random", "Least Confident" and "Incorrect longest word" stand for the random, the least confident word and the longest incorrect word criteria, respectively.

Supervised Words	Recognition	User Supervision	Constrained Search
Random	27.0	25.6	25.8
Least Confident	27.0	22.0	20.8
Incorrect longest word	27.0	19.9	18.5

2.1.2 RWTH

The goal of Task 4.1 of the transLectures project is to develop a fast constrained search system, allowing the user to transcribe a new corpus online, assisted by a recognition system. The basic idea is to receive an initial automatic recognition, which is then displayed to the transcriber. Using an interactive interface, he can correct the recognized text and the search is then repeated by considering the supervised words. This can be repeated iteratively until the user is satisfied with the result.

Both UPVLC and RWTH have developed their own system for fast constrained search up to M12. In the period ending at M24, the RWTH prefix-constrained search implementation has been extended to allow several user supervisions to improve recognition results. For this purpose, two extensions have been developed. Firstly, the search can be broken down into multiple prefix-constrained searches so as to allow prefixes inside a sentence. If it is desired that a prefix is present somewhere inside a sentence, this sentence is cut at this point and a new search is started, beginning with the prefix. The result of the search is then joined with the rest of the sentence, already recognized at an earlier time. Such an infix inside a sentence can have any arbitrary length down to a single word. This makes it possible for the user to supervise any word or sequence of the current sentence and repeat the recognition process. Lastly, words that are not contained in the lexicon can be used in prefixes (or infixes). If a new word occurs, it is detected automatically, broken down into phonemes using a grapheme-to-phoneme model, and added to the lexicon. To avoid recalculating the language model, all new words are mapped to a special unigram.

While the UPVLC system was already evaluated on the poliMedia corpus, RWTH now repeated the same experiments for the VideoLectures.NET corpus. For the initial recognition the current best English recognition system of RWTH was used as described in D.3.1.2. The original system uses a multi-pass approach, which is not applicable to fast constrained search. Hence, only the first pass was used here leading to a slightly higher word error rate. Furthermore, all experiments were evaluated on the eval-set of the VideoLectures.NET corpus.

As proposed by UPVLC in D.4.1.1 three different scenarios are considered to simulate the user input. After performing the initial recognition recognized words are replaced with reference words from a manual transcription in the following manner: First, words are selected at random. For the second method, the recognized words with the lowest confidence are replaced. This is the most realistic approach since the user will most likely correct those words first. Additionally, this simulates the use case that the system proposes words with the lowest confidence first to be corrected by a human transcriber. Third, the longest incorrect words, chosen by an oracle, are replaced. This scenario is unrealistic but is able to show the maximal achievable performance of the fast constrained search approach.

Table 1 shows the results of the evaluation of all three methods on the VideoLectures.NET corpus. First, the initial recognition is reported as baseline. Next, the words are replaced by the

Table 2: Transcription quality in terms of WER calculated on the VideoLectures.NET eval-set using **multi pass** recognition. The columns "Random", "Least Confident" and "Incorrect longest word" stand for the random, the least confident word and the longest incorrect word criteria, respectively.

Supervised Words	Recognition	User Supervision
Random	21.2	20.1
Least Confident	21.2	17.0
Incorrect longest word	21.2	13.8

described approach and the change of the word error rate is measured. In case of random word choice, the experiments are repeated 10 times to reduce the effect of sampling noise. It can be observed that user supervision of only 10% of all words leads to considerable lower word error rates. While the random approach improves the result by 1.5% the choice of least confident words lowers the error rate by 5% making this a good strategy to propose words to be corrected. The third method, choosing the longest incorrect words, shows the maximal achievable result of 19.9%.

Finally, the fast constrained search is performed, leading to a new word error rate. As Table 1 shows, repeating the search with the user-defined constraints further improves the recognition result by 1.2% for selection based on confidence and 1.4% for the oracle incorrect longest word.

User supervision was also applied to the results of the best multi-pass recognition of RWTH. As shown in Table 2 the results are similar to the single-pass system. The baseline word error rate of 21.2% is lowered by 1% using random replacement while choosing least confident words lowers the error rate by 6%. The best achievable result is 13.8%. Results of fast-constrained search on the multi-pass system will be included in the next report.

Finally the experiments of RWTH confirm the findings of UPVLC regarding user supervision. It can be stated that supervising only 10% of the recognized words greatly enhances the result when using the right strategy, here by choosing the least confident words.

2.2 Constrained Search for Translation

Up until M12, RWTH developed a fast constrained search to deal with two kinds of user input:

- The user inputs an unordered set of words (bag-of-words) to guide the translation process.
- The user inputs an ordered sequence of words to guide the translation process.

Experiments with these types of constraints have revealed problems in their practical applications. If they are imposed onto the decoder as hard constraints, i.e. no output violating the constraint is possible, search often fails and no output translation is produced. On the other hand, if we apply them as soft constraints, i.e. following them is rewarded but not necessary, they are often ignored by the decoder.

During this period (M13-M24), an additional option has been developed:

- The user specifies source-target pairs of phrases to guide the translation process.

Table 3: Translation quality on the development set of the German→English VideoLectures.NET task with the application of constrained search. A user effort of 1% is simulated, and several different selection schemes are compared.

system	**dev**	
	BLEU[%]	TER[%]
baseline	21.2	61.2
sequential annotation	22.2	60.5
random selection	21.5	61.3
+ constrained search	21.4	60.8
confidence measure	21.7	60.7
+ constrained search	21.7	60.7
OOV words	21.7	60.1
+ constrained search	22.3	58.9
+ constrained search across sentences	22.6	58.1

In the first two scenarios, the information provided by the user is limited to the target language. In contrast, the third option allows the user to provide partial translations along with the corresponding source word sequences. This way the decoder can link the target translations to their corresponding source sequences and make more focused use of the user input. Also, it can be applied more easily in practice.

The implementation in the phrase-based decoder of RWTH's open source toolkit Jane [28] is very simple. The source-target pairs specified by the user are added on-the-fly the phrase table of the phrase-based decoder and they are given cheap, fixed costs so that they will be preferred over competing translation options.

To evaluate this method, RWTH ran preliminary experiments with a simulated user effort of 1% on the development set of the English→German VideoLectures.NET task. For scenarios are considered:

- sequential correction
- random selection
- confidence measure
- OOV words only

The first scenario is simply having the user start at the beginning of the data set and stopping after 1% of the words have been corrected. Here, constrained search is not applicable. For comparison, also a random selection of 1% of the words is tested. Next, we apply the combined phrase costs as a simple confidence measure. The phrases with the least confidence are selected for correction. Finally, we limit the user effort to correcting out-of-vocabulary (OOV) words, which are usually left untranslated by the decoder. The final setup is motivated as follows. Words unknown to the translation engine usually remain untranslated and therefore have a strong negative impact on the perceived quality of the translation. Furthermore, in the domain of video lectures, they are often technical terms that are repeated multiple times in a single lecture. Therefore, once a user has provided the translation of a technical term, it can be used by the translation engine to correct all of its other occurrences. As an additional benefit, users can directly observe the positive impact of their work, which may serve as an incentive for them to continue working as collaborative users.

The results in Table 3 show, that it is very hard to beat sequential annotation, which improves the baseline by 1.0% BLEU and 0.7% TER. Neither the random selection nor the confidence measure selection schemes give competitive results, and constrained search has little effect in these cases. Selecting OOV words for user correction seems to be more promising. With the application of constrained search, and by re-using the user corrections across the whole development set, rather than limiting their effect to the current sentence, we reach an improvement of 0.4% BLEU and 2.4% TER over sequential annotation.

The small improvements in BLEU can be explained by the nature of this quality measure. Except for a length penalty, BLEU is only the geometric mean of n-gram precisions for $n = 1, \ldots, 4$. Sequential annotation will invariably improve all four n-gram precisions. However, selecting non-sequential words for correction will in most cases only affect the 1-gram precision, which has a much smaller impact on the overall BLEU score. Therefore, TER is the more suitable measure for this kind of user interaction.

3 Intelligent Interaction for Transcription

In the transLectures project, the process of automatically recognizing VideoLectures.NET and poliMedia is expected to be improved by collaborative users. Given that user supervisions are limited and time-consuming, the transLectures project places special emphasis on intelligent interaction between the user and the system. This intelligent interaction is aimed at efficiently managing user effort in order to maximize improvement to transcription quality. The complete set of transcriptions is not intended to ever be fully reviewed since only sporadic user supervision is expected.

3.1 Intelligent interaction approach

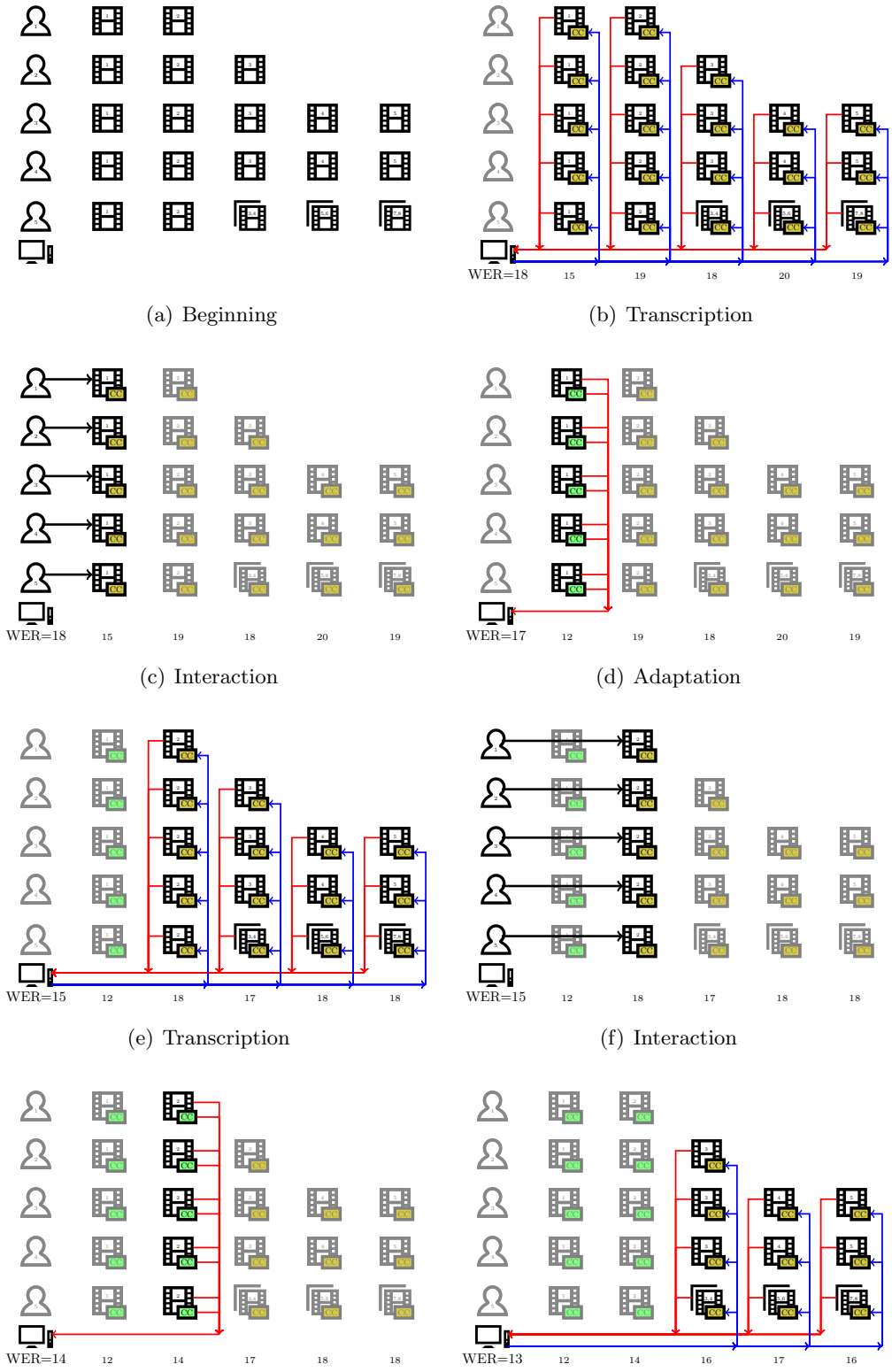
In the M1-M12 period, the UPVLC developed a novel approach for the efficient transcription of video lectures based on limited user effort (a full description of this approach can be found in D4.1.1). Briefly, this approach involved the interactive transcription of a document using an ASR system built from scratch. Before the interactive transcription stage, users had to fully transcribe some documents in order to train the initial ASR system. However, this was unrealistic within the transLectures context because a competitive ASR system was already developed in WP3. In order to address this gap, the UPVLC has proposed a new approach for intelligent interaction (II), similar to that proposed in M12, but which assumes that there exists an initial ASR system. For the sake of clarity, we will describe this new approach making use of the process illustrated in Figure 1.

Let us assume that a set of new video lectures recorded by different speakers needs to be transcribed. Figure 1(a) depicts this situation where there are 23 videos from 5 different speakers (this corresponds exactly with the distribution of videos in the poliMedia test set defined in Task 6.1). These videos are split into 5 blocks of similar duration (blocks in the figures are represented in columns). It must be noted that the organization of the videos into blocks responds to adaptation purposes, since models are retrained after the supervision of each block is carried out. For this reason, the number and size of the blocks is an aspect that must be decided on in order to maximize performance of adaptation techniques. Also, video lectures from one speaker must be distributed sequentially along the blocks to ensure optimal performance of adaptation techniques. The organization of videos adopted in Fig. 1 corresponds exactly to the distribution used in experiments of this task but it is worth mentioning that other different organizations could have been used.

Secondly (Fig. 1(b)), transcriptions ("CC" in the figure, abbreviated from "closed captions") for each video are automatically produced using the ASR system available. Within the transLectures context, this initial ASR system would correspond to the best adapted system developed in WP3 for transcription. In Fig. 1(b) WER shows the transcription quality that could be achieved without user interaction. In Fig. 1, WER is displayed by block (smaller font size) and overall (larger font size).

Thirdly, II is applied to supervise the automatic transcriptions produced in previous steps (Fig. 1(c)). In the II approach, collaborative users devote a limited effort to supervise a given percentage of words of the automatic transcription[1]. User effort is optimised by ordering the speech segments selected for supervision from lower to higher reliability based on confidence measures (CMs). Note that user effort is measured in terms of the number of supervised words, and not in terms of the time required to supervise them, since this time depends on other factors such as user interface design or the individual user's abilities. Also note that user corrections after this first block has been supervised produces reasonable WER improvements in this block (and consequently in overall WER).

[1] In Fig. 1(c) we have simulated a scenario where each user supervises their own videos. However, other scenarios are possible in which one collaborative user could supervise multiple videos or one video could be supervised by multiple users.



(a) Beginning (b) Transcription

(c) Interaction (d) Adaptation

(e) Transcription (f) Interaction

(g) Adaptation (h) Transcription

Figure 1: Intelligent interaction for transcription.

Once this first block of transcriptions has been partially supervised, both the supervised and the high-confidence segments of these transcriptions are used to adapt the underlying system models (Fig. 1(d)). The next step is to automatically re-transcribe the non-supervised blocks using these adapted (and hopefully improved) system models (Fig. 1(e)). Note that in Fig. 1(e) we see that improvements in WER for all blocks are achieved using these adapted models. Also note that some errors remain in supervised blocks as videos are partially supervised.

The process of partial user supervision (Fig. 1(f)), model adaptation (Fig.1(g)) and re-transcription (Fig. 1(h)) is repeated until all available user effort has been invested. As a result of this process, automatic transcriptions have been improved by distributing user effort efficiently in the supervision of lower confidence parts of transcriptions. Moreover, automatic transcriptions have been also improved by using adapted models based on user corrections.

3.2 Summary of progress

The II approach has been evaluated simulating that the whole poliMedia test set has to be transcribed (poliMedia corpus statistics can be found in D3.1.2, Section "Databases used in transLectures"). The poliMedia test set has been split into 5 blocks as has been shown in Fig. 1. The main statistics about this distribution of videos into blocks are shown in Table 4. The best adapted UPVLC Spanish ASR system developed in WP3 has been used as initial system.

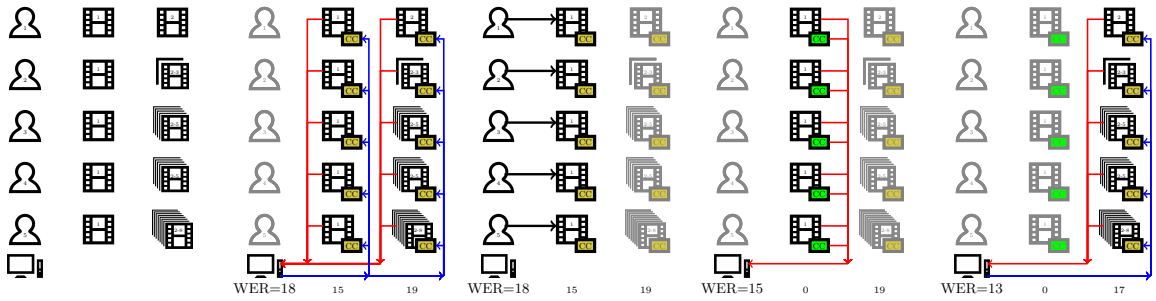
Table 4: Distribution of the poliMedia test set into blocks to evaluate intelligent interaction. For each block we show: time duration, running words and WER [%] (for M12, M18 and M24). We also indicate totals.

	Block					
	1	2	3	4	5	Total
Duration	51'01"	49'57"	39'28"	31'26"	35'14"	3h 27' 6"
Running words	7646	7471	5998	4232	4764	30111
WER (M12)	22.1	24.6	22.7	22.1	22.3	22.9
WER (M18)	21.5	23.4	22.6	21.1	21.2	22.1
WER (M24)	17.2	19.5	20.0	18.7	19.0	18.7

The II approach has been comparatively evaluated against a baseline interaction approach called Batch Interaction (BI). Figure 2 illustrates the process of BI. Figures 2(a) and 2(b) are exactly the same as Figures 1(a) and 1(b), respectively, since, in both approaches, the initial automatic transcriptions are produced in the same way using the best available ASR system. However, in the BI approach, user effort is invested to supervise full transcriptions from the beginning and until there is no user effort left to invest, as would typically be the case in post-editing scenarios. This has been reflected in Fig. 2(c) by considering that the first block is fully supervised by users, which corresponds to applying a user effort of 20%. Note that after user supervision of this first block is carried out, perfect transcriptions of this block will be produced. Consequently, the resulting WER for this block will be zero and the overall WER will also be improved. Once user effort has been completely invested, the perfect transcriptions of the supervised block are then used to adapt the underlying models (Fig. 2(d)). Similarly to II, the resulting updated models are ultimately used to re-transcribe the remainder of videos improving overall transcription quality (Fig. 2(e)).

The II and BI approaches have been performed at two supervision efforts: 5% and 10%. These efforts are equivalent to the percentage of words that the user has to supervise. In the case of II, the words to be supervised are distributed proportionally along the different blocks. (i.e. 5% of words per block in the case of 5% of total effort).

In II, in a given block, the supervision of words is performed from lower to higher confidence until the effort devoted to this block has been invested. The ASR system is fully retrained



(a) Beginning (b) Transcription (c) Interaction (d) Adaptation (e) Transcription

Figure 2: Batch interaction for transcription.

with the words supervised in the current block, and then the following block is recognized. Additionally, a CS step is carried out once the supervision of all blocks is finished. Note that CS can only be applied in II since constraints for each sentence (if any) are provided by the user during the II process.

Figure 3 depicts the progress of II techniques from M12[2]. It shows that II clearly outperforms BI. Comparatively better transcription quality is achieved following the II approach than BI at same level of user effort. This reflects that CMs are good predictor of which words have to be supervised. Moreover, a higher user effort in II produces more significant improvements than BI. In the case of II, double the user effort produces a reduction of about 3 points in WER whereas in BI the reduction is only about 1.5.

The error reduction of each approach is directly related to the percentage of incorrect words supervised by the user. For instance, in BI, errors are corrected uniformly as they appear in the transcriptions from the beginning. This means that the percentage of incorrect words corrected in BI is approximately equal to the percentage of error in the videos. In contrast, in II, the percentage of incorrect words corrected is higher, as CMs selects incorrect words more efficiently. Specifically, in II the percentage of incorrect words corrected are: 74% and 68% for the 5% and 10% experiments, respectively. As observed, the improvement in WER of II is directly related to these figures since 74% of 5% of words is 3.75, which is more or less the total points improvement achieved in II.

As observed, differences in WER between II and BI are higher in M18 than in M24. This is an effect produced by the use of a much better baseline ASR in M24 (Best WP3). Note that comparatively an important improvement in WER was achieved between M18 and M24 in baseline ASR performance with respect to the improvement achieved between M12 and M18 (differences in WER between M12, M18 and M24 for each block and in total can be found in Table 4). Better baseline ASR performance means a lower number of incorrect words. Thus, detection of incorrect words is a more difficult task for II in M24 than in previous months.

Regarding CS (developed in Task 4.1, Section 2.1.1), both implementations (at word and phrase level) have been comparatively evaluated in M24 and very minor differences were found (lower than one decimal point). Unfortunately, phrase-level constraints did not improve word-level CS implementation. A possible explanation for this unexpected behaviour is that differences in how word boundaries are defined in constraints seems to be negligible in the decoding process. Despite this, slight improvements in WER are due to the last step of CS. Results shown in Figure 3 are using CS at word-level. These improvements are higher when user effort is lower. In the case of a user effort of 5%, CS reduces WER by about 0.5 points, whereas in the case of 10% of user effort the reduction is only about 0.1 points. This is expected behaviour since when

[2]Results of M12 do not fully correspond to M12 models, as some changes on the recognition software up to M12 could not be undone.

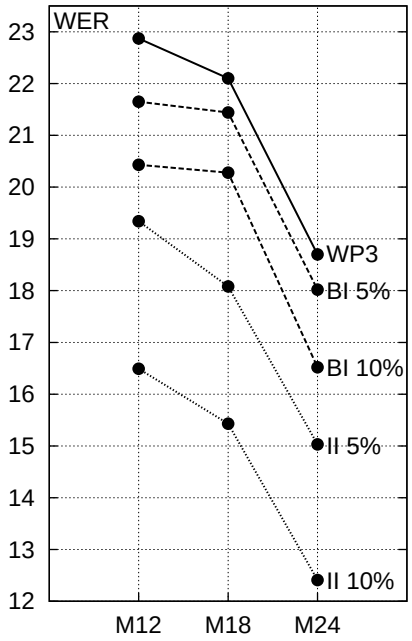


Figure 3: Progress in Intelligent Interaction given in terms of WER. Intelligent Interaction (II) is compared with Batch interaction (BI) at two levels of supervision: 5% and 10%. "WP3" corresponds to the WER obtained by the best UPVLC Spanish ASR system developed in WP3.

more user effort is invested the resulting quality of transcriptions is higher and, consequently, it is more complicated to amend errors applying CS.

As a summary of II performance in M24, let us consider that a collaborative user is able to invest some effort in the supervision of the automatic transcriptions of the poliMedia test set. Following the II approach, the WER of automatic transcriptions can be reduced by nearly four points (from 18.7 to 15.0) by supervising only 5% of words (i.e. 1505 words out of a total of 30111). In the case that 10% of user effort was invested (i.e. 3011 words), the reduction in WER is greater than six points (from 18.7 to 12.4).

An important conclusion of the results presented is that II offers a very good performance, mainly due to the use of CMs to detect misrecognized words. This conclusion was also reported in D4.1.1 and so work carried out within this M13-M24 period has been devoted to improving confidence estimation. A more detailed description of the developed work is presented in the following Section 3.3. On the other hand, in D4.1.1, work was focused in the use of isolated words as supervision unit. Specifically, the user supervises isolated recognised words and makes the appropriate corrections in case of misrecognitions. This raises an important issue, that of the use of a different supervision unit is such a way that II might be improved. For this reason, within this M13-M24 period, experiments have been carried out to compare performance of different interaction units in II. For more details, please refer to Section 3.4.

3.3 Confidence estimation

At M12, CMs were based on word posterior probabilities computed over word graphs [27]. Within the M13-M24 period, an alternative confidence estimation method has been explored. In this new alternative, confidence estimation is carried out using a naïve Bayes (NB) classifier as described in [22]. The reason for selecting this method is that experimental results, again in [22], showed that the NB combination of different features outperforms CMs based on word posterior probabilities.

Additional experiments have been conducted to empirically compare both confidence estimation methods. This comparison has been carried out by evaluating II exactly in the same

way as has been described in previous sections varying the method to compute CMs. The WER obtained depending on the confidence method that has been used will be informative about which is the best performance method for II. In these experiments, the parameters of the NB classifier have been estimated and optimized using the training and development data of poliMedia (details can be found in Section "Databases used in transLectures" of D3.1.2). Table 5 summarizes the WER achieved when each confidence estimation method is used within II at two effort levels (5% and 10%) in M18 and M24.

Table 5: WER obtained by the II approach using different confidence estimation methods (Posteriors and NB) at two levels of user effort (5% and 10%) in M18 and M24.

	User Effort			
	5%		10%	
	M18	M24	M18	M24
Posterior	18.8	15.2	15.7	12.6
NB	18.1	15.0	15.4	12.4

Results show that NB outperforms posterior probabilities as CMs for II. This confirms previously published results [22]. Note that the results plotted in Fig. 3 for II were obtained using NB and, thus, they are exactly the same as in Table 5. Despite this, WER differences are not very large. The reason to this similar performance is that the NB classifier has been estimated using only one feature. Moreover, this feature is also based on posterior probabilities. More significant improvements can be expected if the NB classifier is estimated using a large number of different kinds of features. This will be further investigated.

As we commented in previous section, most of the improvement achieved is produced by CMs. In order to further analyse CMs behaviour an additional experiment has been conducted. In this experiment, we have simulated that each video is separately supervised following II approach for all possible user effort, i.e. from 0% to 100%. After applying each user effort, the quality of the resulting transcription is measured in terms of WER. In contrast to the previous II experiments, neither adaptation nor CS-step was performed in order to isolate the impact of CMs in the supervision. Figure 4 shows, for each user effort in the X-axis, the percentage of the initial transcription WER which is remaining in each video after supervision (gray crosses). For instance, at 0% of user effort the 100% of the original transcription WER is remaining since transcription is not modified. In opposite, at 100% of user effort 0% of the original WER remains since perfect transcription is obtained. This graph gives the trend on how CMs are able to reduce transcription WER as a function of user effort. To better observe this trend, mean (black line) and standard deviation (red error bar) calculated using all videos is also plotted. Also, a diagonal line is plotted to simulate a random behaviour in which WER is reduced proportionally to the user effort employed. It can be observed that CMs are only effective when the supervision effort is below 20%. In fact, supervision of 20% of the words produces between 60% and 40% of reduction in WER. Once this point is reached, the impact of CMs in WER reduction seems to be negligible.

Additionally, the behaviour of CMs for user efforts up to 20% has been more deeply analyzed. For this purpose, relative WER reductions have been computed after 5%, 10%, 15% and 20% of supervision effort has been applied to each video transcription. Figure 5 shows a four stripe histogram for each video. X-axis shows the initial transcription WER of the lecture. Histograms represent for each stripe the relative reduction in terms of WER that would have been obtained after supervising 5% (red), 10% (yellow), 15% (green) and 20% (blue) of the recognised words. As can be seen, the relative improvement quickly decays once 15% of the words have been supervised. The most important relative reductions in WER are achieved using 5%, 10%, and 15% of supervision effort. The lower the user effort, the greater the impact of the CMs. On the other hand, we also observe unexpected behaviour: CMs performance is not related to the initial WER of the video. In fact, it can be observed that the largest relative improvements are achieved in lectures with the lowest WER.

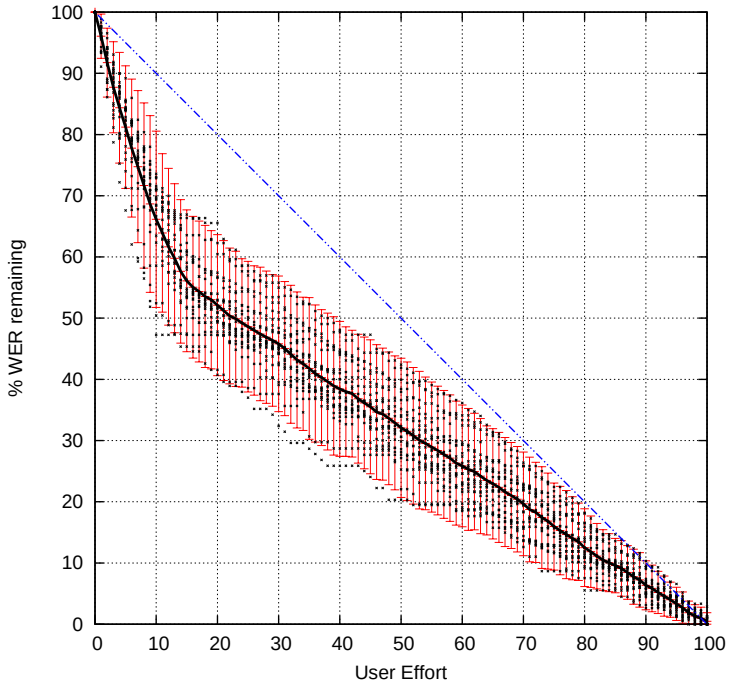


Figure 4: Percentage of remaining WER (from the initial transcription WER without supervision) after II for all possible user efforts on the poliMedia test set. Results are expressed for each video (gray crosses), along with the mean (black line) and standard deviation (red error bars) calculated using all videos.

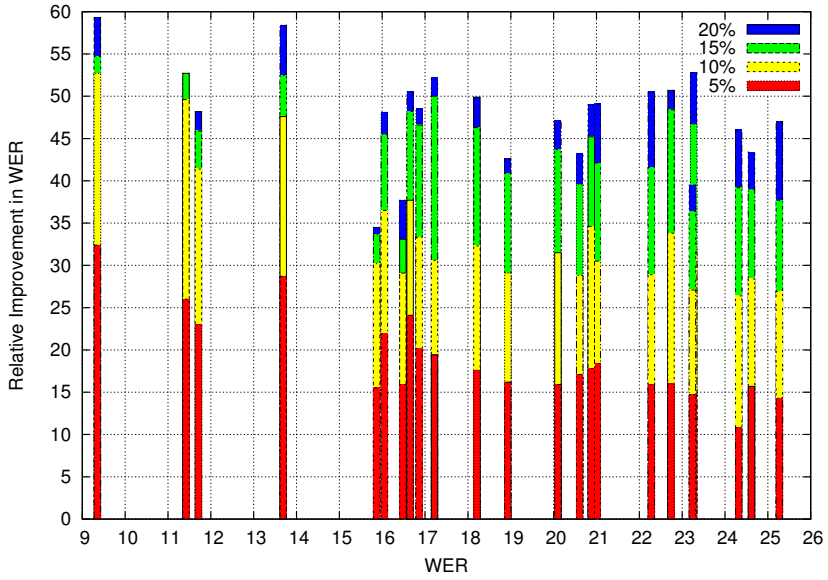


Figure 5: Relative WER reduction [%] for each lecture when supervising: 5%, 10%, 15% and 20% of the recognised words. Lectures are ordered in the X-axis according to their original WER.

3.4 User interaction units

Up until M12, only isolated words were considered as user interaction units. Within this M13-M24 period, II has been evaluated using new interaction units such as sequences of consecutive words (i.e. phrases), and sentences. The motivation behind this work is to evaluate the behavior of II when interaction units are changed from words to phrases or sentences. In case of word supervision, user supervises a single word without any kind of context. However, II can be more suitable to the user if interaction units provide more contextual information and better comprehension. This is the case of sentences which are typically used in many document transcription tasks.

Phrase and sentence supervision is performed also according to CMs. In case of phrases, all combinations up to four consecutive words were considered. CMs at phrase-level were estimated as the sum of individual CMs at word level. It must be noted that each time a phrase is supervised, all phrases that are contained within the supervised phrase are ignored in order to avoid the supervision of the same words. In case of supervision at sentence-level, CMs are estimated as the geometric mean of individual CMs of each word in the sentence. In the same way, sentences are ordered according to confidence and presented to the user from lower to higher confidence.

Experiments were carried out following the experimental setup used in D4.1.1 (assuming initial empty models) since the new experimental setup had not yet been designed. Table 6 reports the quality of the end transcriptions in terms of WER after the 96-hour Spanish transLectures training set has been supervised according to the different interaction protocols (word, phrases or sentences). These experiments took an user effort of approximately 20%. In the case of word- and phrase-level supervision, CS is applied after the user has completed the supervision process. Results show that words are more effective than both phrases and sentences as interaction units. This can be explained by the degradation of CMs when moving from word to phrase or sentence level. This effect is particularly significant in the sentence-level approach, which only slightly outperforms BI. Note that CMs in the phrase- and sentence-level approaches are obtained by adding together individual word confidences. These results confirmed that words are the best performance interaction unit and, in consequence, all II experiments using the new experimental setup were carried out using words as interaction unit.

Table 6: Final WER of the automatic transcriptions after 20% of the words of the complete Spanish transLectures training set are supervised using the different interaction approaches.

	BI	II (Word)	II (Phrase)	II (Sentence)
WER	25.0	16.3	19.7	24.0

3.5 Planned work for the M25-M36 period

Confidence estimation has proved to be the most relevant influence in the improvement of II. For this reason, in the next M25-M36 period, work will be focused in the improvement of confidence estimation. With this purpose, different actions have been planned:

- Improve NB classifier using a larger set of features.
- Use speaker adaptation in confidence estimation.

Meanwhile, UPVLC plans to improve system adaptation within the II approach. For instance, the recognition parameters could be adjusted differently for each speaker, or a better system adaptation could be performed by a speaker-based selection of samples.

4 Intelligent Interaction for Translation

In the transLectures project, in addition to the automatic transcription of VideoLectures.NET and poliMedia, a full set of translations for these transcriptions is produced. Similarly to the transcriptions, translations are intended to be improved by collaborative users. The task of supervising translations is even harder than that of supervising transcriptions since it is a much more time-consuming process than transcription. For these reasons, the emphasis on intelligent interaction in the transLectures project is not only focused on transcription but also on translation. As with automatic transcriptions, the main objective of intelligent interaction for translation is to get the best possible set of translations in exchange for a fixed amount of human effort.

4.1 Intelligent interaction approach

In the period leading up to M12, UPVLC developed a novel approach for the efficient translation of video lectures based on limited user effort (a full description of this approach can be found in D4.1.1). However, this approach was evaluated following an experimental setup which was not perfectly in accordance with the transLectures context. The reason behind this decision was to evaluate II for translation within an scenario in which a very large number of automatic translated sentences have to be supervised. Note that VideoLectures.NET and poliMedia test sets are very small corpora (1.3K and 1.1K sentences, respectively). With the purpose of increasing the number of sentences, VideoLectures.NET and poliMedia training sentences were included as part of the corpora on which to evaluate the proposed approach. This produced as a result that SMT systems developed in WP3 could not be used as initial systems and external resources had to be employed to develop them. In order to solve this mismatch and evaluate II within a more appropriate scenario given the transLectures context, a new experimental setup has been defined within this M13-M24 period.

Two main differences can be pointed out between the two experimental setups. On the one hand, in the new evaluation proposal the best adapted SMT systems developed in WP3 will be used as initial systems in this task. On the other hand, evaluation will be carried out on the official scientific tests of transLectures. Although certainly scientific tests are not very large, this evaluation will allow us to measure improvements provided by II with respect to the baseline performance of WP3 SMT systems. As a result, this new experimental setup will be more in accordance with the transLectures context in which II techniques are applied to improve performance of baseline systems developed in WP3.

As a reminder of the II translation approach proposed in D4.1.1, we will describe it here making use of the process illustrated in Figure 6.

Let us assume that a set of new video lectures from different speakers have to be translated. Figure 6(a) depicts this situation where there are 23 videos from 5 different speakers (this corresponds exactly with the distribution of videos in the poliMedia test set defined in Task 6.1). At the beginning, we will assume that transcriptions (CC in the figure) for each video have been produced in some way. Note that these transcriptions may contain errors if they have only been partially (or even not) supervised.

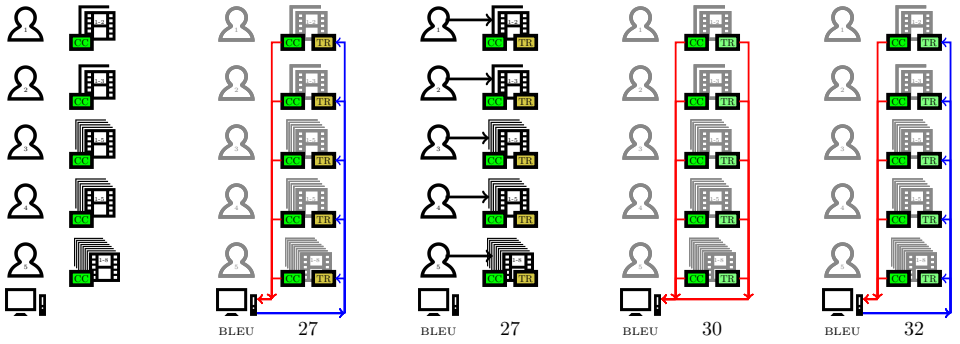
Secondly (Fig. 6(b)), translations (TR in the figure) are automatically produced for each video by translating transcriptions using the best SMT system available for the pair of languages aim of translation. As has been mentioned, this initial SMT system would correspond to the best adapted system developed in WP3. In Fig. 6(b) BLEU [21] reflects the translation quality that could be achieved without user interaction.

Thirdly, II is applied to supervise the automatic translations produced in the previous step (Fig. 6(c)). As in the case of II for transcription, supervision is carried out by collaborative

users who devote a limited effort to supervising a given percentage of translated words[3]. User effort is efficiently employed by supervising translation segments from lower to higher reliability based on confidence measures (CMs). Note that BLEU improvements can be expected after user supervision of automatic translations is carried out (from 27 to 30 in the figure).

Once translation quality has been improved after user supervision, supervised segments of these translations are used to adapt SMT models (Fig. 6(d)). As a final step, non-supervised translations are automatically regenerated (and hopefully improved) using these adapted system models (Fig. 6(e) in which an improvement of BLEU is reflected).

As can be observed, the process depicted in Fig. 6 is very similar to that presented in Fig. 1 for II for transcription. The only difference between the two methods presented is that in II for translation videos are not split into different blocks for supervision. This is a minor issue since, in fact, the same process of user interaction (Fig. 6(c)), model adaptation (Fig.6(d)) and translation (Fig. 6(e)) could be performed in an identical way following a block-level organization of videos. Indeed, II for translation was evaluated in D4.1.1 following a block-level organization of videos. The decision about whether or not it is convenient to split videos into blocks is subject to adaptation purposes. In this case, Fig. 6 illustrates the process that has been adopted in this M13-M24 period to evaluate the II approach over the poliMedia test set. As the number of sentences in poliMedia test set is too small to adapt the system several times, videos have been organized in a single block.



(a) Beginning (b) Translation (c) Interaction (d) Adaptation (e) Translation

Figure 6: Intelligent interaction for translation.

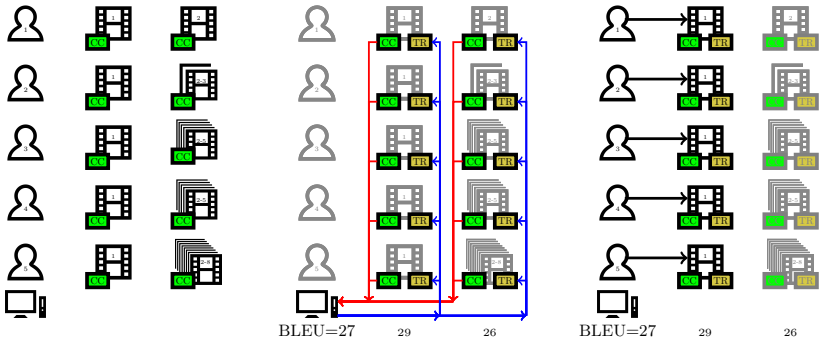
4.2 Summary of progress

As in the case of II for transcription, II for translation has been also comparatively evaluated against batch interaction (BI).

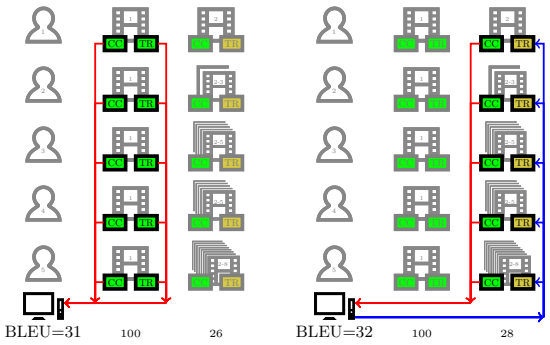
The BI approach is illustrated in Fig. 7. As can be observed, videos are organized into two blocks in BI. This organization responds to the idea that user effort will be employed in the full supervision of translations starting from the first translated sentence and until user effort is completely invested. Thus, the first block will contain exactly the number of translated sentences whose supervision is equivalent to the user effort. In the Fig. 7 a scenario has been simulated in which each user fully supervises one of their videos but this is only for illustrative purposes.

At the beginning of BI, transcriptions (again, "CC" in the figure) of each video are automatically translated ("TR") in the same way as in II using a baseline SMT system (Figs. 7(a) and

[3]Fig. 6(c) simulates a scenario in which each user supervises its own videos although other supervision scenarios are possible as was mentioned in previous section.



(a) Beginning (b) Translation (c) Interaction



(d) Adaptation (e) Translation

Figure 7: Batch interaction for translation.

7(b)). BLEU figures show the translation quality of automatic translations for each block (lower font size) and overall (higher font size). In a third stage (Fig. 7(c)), automatic translations of the first block are fully supervised by users producing perfect translations (this is reflected in Figure 7(d) with a BLEU of 100 for the first block). Note that this is the main difference between II and BI approaches, since in II user effort is employed in the partial supervision of all videos and focused only on the translations with lower confidence. Once user effort has been completely invested, perfect translations of the supervised block are then used to adapt the SMT models (Fig. 7(d)). Finally, the adapted SMT system is used to translate the remainder of videos improving translation quality (Fig. 7(e)).

Both approaches have been evaluated, simulating that the whole poliMedia test set has to be translated (poliMedia corpus statistics can be found in D3.1.2, Section "Databases used in transLectures"). It has been assumed that perfect transcriptions are available for all videos. The best UPVLC Spanish→English SMT system developed in WP3 has been used as the initial SMT system. The II and BI approaches have been performed at four levels of supervision efforts (2%, 5%, 10% and 20%). These efforts are equivalent to the percentage of words that the user has to supervise. II has been evaluated at two different interaction levels: sentence-level (IIS) and word-level (IIW). In the case of IIS, user interaction consists of the supervision of whole sentences. In this case, user interaction is simulated by simply replacing the supervised sentence by its reference translation. In the case of IIW, user interaction consists of the supervision of isolated words. In this case, given that simulation of user interaction is not so trivial as in the case of supervision at sentence-level, an illustrative example has been depicted in Fig. 8. As is shown in the figure, word alignments are used to simulate which words would be replaced by the user to correct the erroneous translated word. Specifically, the supervised word ("gets" in the example) is replaced by the reference words which are aligned with the same source words that have been aligned with the supervised word. As a result in the example, the word "gets" would be replaced by the words "are they filed before". In this way, supervision of word "gets" would

convert the translated sentence "to whom it gets?" into "to whom it are they filed before?". In both cases, supervision units (sentences or words) are supervised from lower to higher reliability based on confidence measures (details about confidence estimation can be found in following sections).

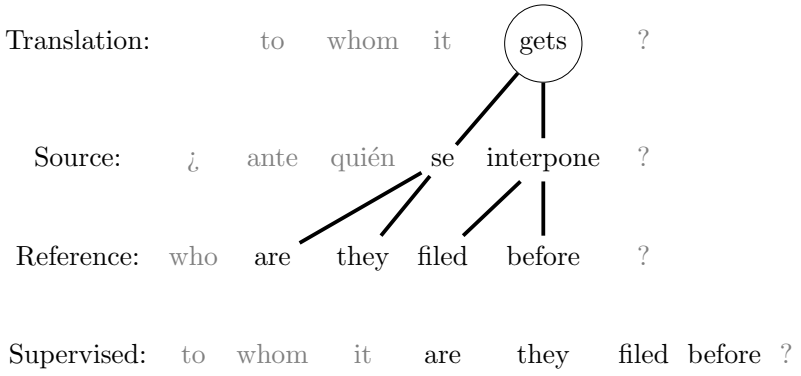


Figure 8: Example of user interaction at word level.

Figure 9 depicts the progress of the task from M18[4]. For simplicity, only results using 5% and 10% of supervision efforts have been plotted (results for all supervision levels can be found in following sections). Results for IIW are given only in M24 since this kind of interaction was implemented within the M18-M24 period.

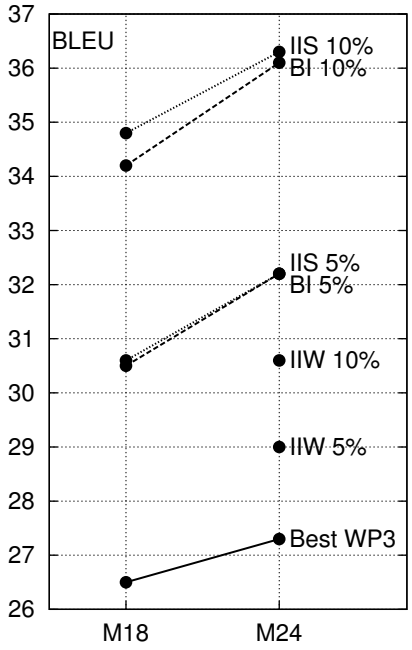


Figure 9: Progress in Intelligent Interaction given in terms of BLEU. Intelligent Interaction (II) is compared with Batch interaction (BI) at two levels of supervision (5% and 10%). II is performed using two kinds of interaction units: sentences (IIS) and words (IIW). "Best WP3" corresponds to the BLEU obtained by the best WP3 UPVLC Spanish→English SMT system.

Regarding interaction at sentence-level, it can be appreciated that II and BI present very similar performance. More precisely, IIS produces some slight improvements over BI only when 10% of user effort is employed. This similar behaviour seems to indicate that confidence measures are not a good predictor of the correctness of the sentence. However, a more in-depth

[4]M12 results are not plotted since they were performed using a different experimental setup as has been mentioned.

analysis of the results obtained in M18 concluded that IIS has no potential to improve BI. In this analysis, translated sentences were scored by an Oracle computing the translation error rate (TER) [23]. These scores were used as an objective measure of the correctness of the translated sentences and, thus, sentences were supervised from higher to lower TER. As a result, only improvements of around one point of BLEU were found between BI and Oracle approaches. This fact explains that supervision of sentences following the order of appearance in the videos (BI) or processed from lower to higher confidence (IIS) yield very similar results. One possible explanation might be that the sentence level is too coarse, despite being a more natural user interaction unit for translation. Note also that even the worst translated sentences have sections that are correct, which consume user effort without leading to any improvement.

A possible solution to improve II performance could be to focus user effort only on the erroneous segments of translations in similar way as is done in transcription. In this way, user effort will be employed more efficiently than in the case of supervision at sentence-level where correct parts of translations are also supervised. With this purpose, alternative interaction units (i.e. words, segments of consecutive words, etc.) should be explored. Experiments using isolated words as interaction unit have been carried out in the M18-M24 period. In this case, word-level CMs are used to supervise words from lower to higher confidence. Figure 9 shows the resulting BLEU which is achieved after II is applied using words as interaction units. Although results are worse than using sentences as the interaction unit, user simulation influences this comparatively worse behaviour.

User simulation in the case of words is not as effective as in the case of sentences. Supervision at sentence-level is simulated by replacing whole translated sentences by the reference sentences. This means that user supervision produces the best possible output and reordering, deletions, substitutions and insertions are produced in the same supervision action. However, in the case of words, although reference words are replaced by mistranslated words, sometimes positions where are located reference words are not so optimal with respect to the position in the translation. This is an important issue that needs further investigation in order to better evaluate performance of II at word-level.

In order to more precisely compare II at word-level to BI, a BI variant was implemented in which user supervision is performed without allowing re-ordering of words. This variant has not been plotted in Fig. 9 since it is somewhat unrealistic to assume that re-ordering of words is not allowed in full supervision of sentences. In any case, this has been done to evaluate whether II at word-level outperforms BI under similar user interaction conditions. Results show that very similar performances are achieved between II at word-level and BI (a full description of these experiments can be found in Section 4.4). This means that CMs at word-level are not helpful for detecting mistranslated words.

Finally, we assessed the impact of adapting the models using the supervised sentences in order to translate the remaining unsupervised sentences. We found that the quality improvement obtained by adapting the system (language and translation models) was not statistically significant. A possible explanation is that the corrected sentences account for less than 0.1% of the corpus which, in practice, barely changes the models.

In the following sections, a more detailed description of the work developed in this M13-M24 period is provided, differentiating by type of interaction unit.

4.3 Intelligent interaction at sentence level

II at sentence level is carried out by supervising sentences from lower to higher reliability based on CMs. In D4.1.1 confidence measures at sentence level were computed using sentence posterior probabilities as has been proposed in the literature [25]. Table 7 shows an example to illustrate how sentence posterior probabilities are computed. This calculation is performed using the N

Table 7: Example of the 5 best translations of Spanish sentence "Aquí tenéis un ejemplo". "SMT Scores" are the figurative values provided by the SMT system to rank hypotheses. "Posterior" gives the sentence posterior probabilities computed based on SMT scores.

N	Translated Sentence	SMT Score	Posterior
1	Here is an example	35	0.35
2	Here you can see an example	30	0.30
3	Here you see an example	20	0.20
4	One example here is	10	0.10
5	Here example is shown	5	0.05

Table 8: Results in terms of BLEU for the different sentence-based interaction strategies for translating the Spanish→English transLectures test at several supervision levels using M18 and M24 SMT systems.

	M18				M24			
Supervised Words	2%	5%	10%	20%	2%	5%	10%	20%
Initial (WP3)	26.5				27.3			
BI	28.1	30.5	34.2	42.1	29.5	32.2	36.1	44.0
II	28.2	30.6	34.8	42.8	29.4	32.2	36.3	44.4
II (Oracle)	29.3	31.3	35.1	43.5	29.5	32.1	36.3	44.5

most probable translated sentences (known as N-best list). In this way, as is shown in Table 7, the posterior probability of a sentence is computed by dividing its SMT score by the total amount of all SMT scores in the N-best list.

In this M13-M24 period this approach has been deeply analyzed following the new experimental setup described previously in Section 4.1. As a way to measure the potential of supervision at sentence level, an Oracle approach has been also tested. Oracle scores sentence quality by computing translation error rate (TER) [23] based on the automatic and reference translations. Then, the worst scored sentences (in order of priority) are presented to the user for their supervision. Although this Oracle approach is somewhat unrealistic, it allows us to assess the potential for improvement within a sentence-level interaction protocol.

These interaction techniques have been compared with varying amounts of fixed human effort (2%, 5%, 10% and 20%). As was mentioned in the previous section, II experiments have consisted of simulating that the whole poliMedia test set have to be translated. The best UPVLC Spanish→English SMT systems developed in M18 and M24 have been used as initial SMT systems.

Table 8 shows the results achieved by the three interaction techniques in terms of BLEU. It is observed that, even with an Oracle that perfectly scores the worst sentences, II techniques yield only slight improvements. For instance, in the case of 20% user effort, a modest improvement can be achieved by II (Oracle); while in the more realistic II framework, a negligible improvement is obtained with respect to the BI approach. The fewer the words supervised by the user, the smaller both the potential for improvement and the improvement itself become. In view of these results, it seems to be useless to consider sentences as supervision units. Another issue that this raises is that constrained search cannot be exploited in the case of sentences. As a consequence of this study, supervision at sentence-level should be dismissed in favour of other interaction units which offer greater scope for improvement.

4.4 Intelligent interaction at word level

II at word level involves computing word-level confidence measures. To this end, 3 different word-level confidence measures based on N-best lists have been computed. These features are based on word posterior probabilities and they were proposed in [25].

The method for computing word-level CMs using N-best lists is very similar to that described for the case of sentence-level confidence measures. Basically, given a translated word e, its posterior probability is computed by adding the posterior probabilities of all sentences in the N-best list which contain e. Several word-level posterior probabilities can be computed by using different criteria to decide if e is contained within a sentence. We have used three different criteria: Levenshtein position (Lev), Fixed position (Fix) and Any position (Any). We will make use of the example in Table 7 to illustrate each one of this methods by computing the posterior probability of the word "example". In the case of Levenshtein position criterion, posterior probability of the word "example" is computed by adding the posterior probabilities of the N-best sentences in which the word "example" is Levenshtein-aligned to itself in the 1-best. In Table 7 this would mean adding the sentence posterior probabilities of the first third sentences, resulting in a posterior probability of 0.85. In the case of Fixed position criterion, the sum is performed by considering the N-best sentences that contain the word "example" exactly in the same position as in the 1-best. In Table 7 this would mean adding only the sentence posterior probability of the first sentence, giving a posterior probability of 0.35. In the case of Any position criterion, the sum is performed by considering the N-best sentences that contain the word "example" in any position. In Table 7 this would mean adding the sentence posterior probabilities of all the N-best sentences, giving a posterior probability of 1.0.

Additionally, we use another confidence measure based on the translation Model 1 proposed by IBM in [2]. This confidence measure has demonstrated to be very useful to detect mistranslated words [25]. Given a translated word e, IBM1 confidence measure is defined as:

$$\text{IBM1}(e) \quad = \quad \max_{0 \leq j \leq J} p(e|f_j) \tag{4}$$

where $p(e|f)$ is the lexicon probability based on IBM model 1, f_0 is the empty source word, and J is the source length.

As in sentence-level experiments, an Oracle approach has been also tested. The Oracle selects translated words that are not in the reference for supervision. It allows us to assess the potential for improvement within a word-level interaction protocol. These interaction techniques have been compared with varying amounts of fixed human effort (2%, 5%, 10% and 20%) and assuming that the entire poliMedia test set has to be translated. The best UPVLC Spanish→English SMT systems developed in M24 have been used as initial SMT systems.

Table 9 shows the BLEU achieved after II is performed using each proposed method of confidence estimation. BI is computed without allowing re-ordering in the supervision of sentences to be more adequate with evaluation conditions in word-level experiments. Results show that II does not significantly outperform BI for any confidence estimation method. This means that CMs are not effectively detecting mistranslated words. Meanwhile, the Oracle approach seems to indicate that there is no significant room for improvement, particularly for the lower user efforts.

Table 9: Results in terms of BLEU for the different word-based interaction strategies for translating the Spanish→English transLectures test at several supervision levels.

Supervised Words	2%	5%	10%	20%
Initial (WP3)	27.3			
BI	28.2	29.1	30.7	34.0
II(Lev)	28.0	28.9	30.5	34.1
II(Fix)	28.1	29.0	30.3	33.2
II(Any)	28.0	29.0	30.6	34.0
II(Ibm)	28.3	29.2	30.2	32.4
II(Oracle)	28.6	30.7	33.9	39.8

4.5 Planned work for the M25-M36 period

An important conclusion of the work carried out within this M13-M24 period is that further research is needed to make supervision units smaller than sentences efficient in terms of the user effort required. In this regard, II at word-level has been explored as a first attempt. Although results at word-level have not been very encouraging, different actions have been planned to improve this kind of interaction:

- Improve simulation of user interaction at word-level, performing a more appropriate evaluation of the techniques.
- Improve confidence estimation by using a combination of features.

All these actions will be used as solid work to extend user interaction to phrase level (sequence of consecutive words) since it seems to be a more appropriate unit for interaction in translation.

Additionally, other actions are planned:

- Apply constrained search techniques for translation developed in Task 4.1 (see Section 2.2).
- Improve model adaptation by using non-supervised techniques.

5 Incremental Training for Translation

Overview

During the M13-M24 period, XRCE has focused on an approach to incremental training for translation consisting in "quick-update" short-cycles updates based on "mini-batches" of new human translations. We perform alignment and phrase-extraction from the mini-batches and combine the resulting phrase-tables with the tables obtained by slower long-cycles updates based on batch-training. Different configurations are evaluated and these appear to be superior to an incremental training model based on Incremental-Giza and on suffix arrays. These experiments indicate the promise of the approach for quickly incorporating translation post-editions into the transLectures *training workflow, at least on a daily basis, while performing a full retraining of the models with a slower periodicity, perhaps once a week.*

5.1 Introduction: incremental training through quick-updates

The most straightforward way to comprehensively update an SMT model based on new data is to re-train the model with the entire data that is available at a given time. This kind of training is often referred to as *batch* (re-)training. Such a process is time consuming and intensive in computational resources, especially when large datasets are involved. In consequence, it may not be feasible to run it often enough, resulting with long lags between two batch updates, during which the running system is not up-to-date with the newest possible model.

Incremental training algorithms address this issue by enabling an SMT model update based on the new data rather than retraining the model from scratch.

One way to perform incremental training is by using online versions of the Expectation Maximization (EM) algorithm, that is employed in the alignment step of the SMT model construction. However, this is a relatively new line of research and available mature tools to perform the required updates efficiently are still largely missing. In this section, we take a slightly different approach to incremental training. We explore different configurations of the SMT system that also provide means for utilizing new data in between batch updates. We compare several such configurations, where in-domain data is based on spoken language transcriptions both from transLectures and from other sources, to assess which methods are practically useful for quickly updating the model, especially when the new data belongs to the target domain.

Consider the following setting of an automatic translation system that is either a standalone translator or as part of a larger software system. The system is deployed and is being used, while more training data is becoming available constantly, e.g. through users who provide corrections to the system's translations. To use this data, two kinds of update cycles are employed: (i) a long cycle (e.g., a week), at which end we can perform a *slow update*, which can include re-training, tuning and any other time-consuming tasks; (ii) a short cycle (a day, for instance) in which we wish to carry out a *quick update* consisting of only light-weight tasks that are guaranteed to complete in a timely manner. In these short cycles the model is updated based on the newly obtained data. The goal is to improve the model with respect to the previous slow update, and reflect the received feedback; we do not necessarily expect to obtain as good a performance as the following slow update, but hope to be in the same ballpark. The focus of the current section is in identifying the most appropriate setting for quick updates, both it terms of translation quality and of time. That, with tools that are currently available.

5.2 Background

Incremental training methods provide a principled way for updating an SMT model when more data is received, without generating it again from scratch. In addition to efficiency, such methods hold the promise to reflect updates immediately, without work interruption, and are therefore of major importance in many scenarios.

Incremental training for MT often makes use of an online version of the Expectation Maximization (EM) algorithm [4]. EM is used for the purpose of aligning the bilingual corpus while computing translation probabilities [1]. In *Online EM*, the model parameters are updated after each example or a small set of examples (*mini-batch*), and not for the entire dataset at once. Naturally, online EM is faster than *batch EM*, but is may be less stable.[5]

Ortiz-Martinez et al. [20] use the incremental online EM algorithm [18] to update a standard log-linear model. They apply this in the context of Interactive Machine Translation, where conveying the impression of a highly adaptive system to the user is particularly important. A method for incrementally updating SMT models was also proposed within the SMART project [6]. Features are extracted from post-edited translations, and are added to the model for any phrase above a given sentence-level BLEU threshold. Levenberg et al. [16, 15] use *stepwise EM* for updating the translation model parameters. They use IBM Model 1 [1] with HMM alignments [26], collecting counts for translations and alignments and updating them by interpolating statistics of the old and the new data. We employ and assess an implementation of this algorithm within Moses (Section 5.5).

One useful domain adaptation method is based on mixture models. Training data is divided into components, according to the different domains. A model (either a translation model or a language model) is trained for each component separately and the models are then weighted and combined to a complete model [10, 14]. We use this approach in some of the configurations we assess. However, our goal is different: we focus on the capability to perform the updates quickly. Fortunately, as our results show, these considerations often go hand in hand, and methods that work well for domain adaptation purposes are often useful also for quick updates.

SMT model generation

For completeness, we briefly describe here the main steps in generating a basic phrase-based SMT model, from a parallel corpus to a tuned model. Our description corresponds to the steps as done in Moses [13], but is typical to most phrase-based SMT systems.

- **Preprocessing**: The model generation process starts with preprocessing of the bilingual parallel (sentence-aligned) corpus, including tokenization, lower-casing, and removal of sentences that are, e.g., very long.
- **Alignment**: Following some file preparation steps, GIZA++ [19], an implementation of the IBM Models [1], is applied in two directions (source-target and target-source) to produce word alignment within each source-target sentence pair. A symmetrization of the GIZA bi-directional word alignments follows.
- **Phrase table construction**: Based on word alignments, a *translation model* is generated: lexical (word) translation probabilities are computed and phrases are extracted, scored and stored in a phrase table (PT).
- A **reordering table** is constructed to model position change of phrases between the source and the target.[6]

[5] See [17] for a detailed discussion of the variants of online EM algorithm.

[6] Phrase extraction is also needed for this step; we chose to include it within the translation model generation step since, as explained later, we do not update the reordering model in this work.

- A **language model** (LM) is generated from the target side of the parallel corpus and possibly additional target language monolingual data.
- Lastly, **tuning** takes place in order to optimize the weights of individual scores (features) in the complete model.

Large phrase tables, reordering tables and language models that cannot fit into memory are often binarized for quick loading and access at translation time. Yet, binarization is not feasible when very large tables are concerned. Reducing the size of the tables through filtering based on a given test dataset is not practical in real world applications, and is slow to process as well, as it depends on the size of the tables.

Of the above, alignment is the most time consuming step; phrase table construction may also requires a substantial amount of time, especially when binarization is performed; tuning involves multiple iterations (typically over 20) in which a development set is being translated and evaluated, and is therefore a highly time-consuming task. Indeed, some of the steps can be parallelized, yet not all. For instance, in MGIZA [11], a multi-threaded version of GIZA++, sentences-pairs are aligned in parallel, saving a substantial amount of time; still, parameter estimation is based on counts that are accumulated from all aligned sentences, and is not parallelized. What is often referred to as *batch training* consists of all the above steps applied to the entire data.

Figure 10 shows the relative time required to complete each task, based on an experiment we conducted, with 1 million sentence-pairs for training and 1,000 sentence-pairs for tuning. Both datasets were taken from the Italian-English corpus of Europarl version 7 [12]. As elapsed time depends on the specific machine and its load at the time of measurement, we use the Unix `time` command for obtaining duration information. We look at the accumulated CPU time, which is roughly equivalent to running on a single CPU. For intuition, the alignment task used up approximately 21 CPU hours, which corresponded to about 6 actual hours when running MGIZA with 4 cores. For comparison, under the same machine configuration, alignment of 2,000 sentences took 2.5 CPU minutes, and 10,000 sentences required less than 13 minutes. This experiment was performed on a 64 bit Linux machine, with four 2.67GHz cores and 50GB of RAM.[7]

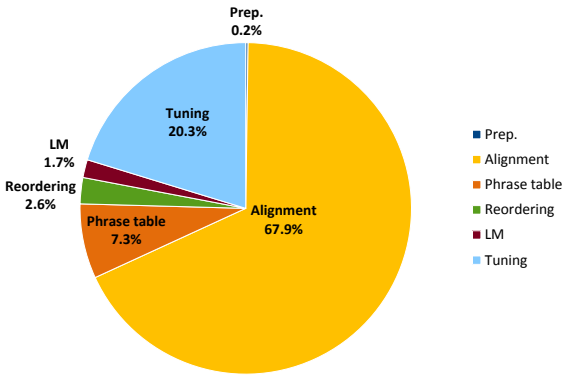


Figure 10: Percentage of the time required by each task of the phrase-based model generation. The time shown here includes the binarization of the corresponding model.

5.3 Quick update configurations

Let's recall the scenario we take interest in: An SMT system is trained based on a large out-of-domain corpus and is meant to be used on a different type of dataset, namely spoken-language

[7]The Moses manual provides details of another timing experiment. There, similarly to ours, alignment is by far the most time-consuming task. Note that we divide the steps somewhat differently.

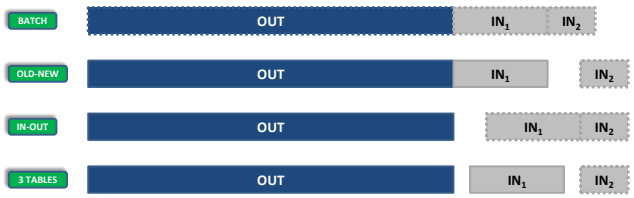


Figure 11: Phrase tables in the different configurations. OUT denotes out-of-domain data; IN_1 is in-domain data previously obtained, and IN_2 is the in-domain data we have just received and wish to use to update the system with. The dashed lines designate the data that needs to be processed in each update cycle.

texts. The translation service is made available and gradually in-domain data is flowing in, e.g. from post-edition. With this data we wish to update the system in the most efficient and effective way. We expect best translations to be produced when we use all the data we have at our disposal; at the same time, we do not wish to carry out intensive processes unnecessarily. We therefore carry out batch updates periodically (long-cycles), and in the interim perform quick, short-cycle updates using the newly obtained data. We wish to identify the most useful configuration – in terms of time and translation quality – for performing short cycle updates. For that purpose, we examined the following configurations for quick model updates. Each has its pros and cons, as discussed below. Figure 11 depicts the phrase table settings of these configurations.

1. **Old-New**: In this setting we use two phrase tables. We maintain all previously obtained ("old") training data, both in-domain and out-of-domain, in one phrase table and the newly obtained data ("new") in a second phrase table. To perform an update, we only need to preprocess and align the new data on its own and generate a phrase table from it. This is therefore a very quick way to perform updates.

2. **In-out**: This setting uses two phrase tables as well, but now the out-of-domain data is maintained in one table and the in-domain data in another table. The idea is to allow better model tuning by letting the tuning method give preference to the in-domain table. The drawback is that all in-domain data needs to be processed at every short-cycle update, implying a longer process. As long as in-domain data is limited, this is not an issue. On the contrary, it can contribute to improved alignment quality and phrase table statistics.

3. **3-tables**: When in-domain data accumulates, the In-out setting may become too slow. We therefore assess another setting that can potentially combine the benefits of the two above configurations. Here, we use three phrase tables: one for out-of-domain data and two for in-domain. The first among the in-domain tables is used for all previously obtained in-domain data, and the second for the newly obtained data. This way we achieve both separation of in- and out-of-domain data and a quick processing of the new data.

4. **Batch**: This is a standard setting for phrase-based SMT model generation, used here for comparison. The entire training data is concatenated and used together, and a single phrase table is produced. One potential advantage is, as above, an improved alignment quality.

In addition to the above configurations, we have experimented with incremental updates via Moses' dynamic suffix array. We describe that in Section 5.5.

Required effort

Table 10 details which task needs to be performed in each configuration, and the amount of work that has to be done. We explain the required effort of each configuration through an example,

whose timeline is presented in Figure 12: We consider a specific point in time of an operational translation system. This system was trained with 1 million out-of-domain sentence-pairs before any in-domain data was available (S_1 in the figure); over time, 10,000 in-domain bi-sentences were received and the system has been already updated with them in a slow update cycle (S_2). Between the previous slow update and the current point in time, 2,000 in-domain sentences have been obtained, and fed into the system (q_{21}); now we receive 1,000 more, and wish to carry out quick update q_{22}.

Figure 12: The timeline of slow and quick updates, as described in the required effort example in Section 5.3. S_i denotes a slow update and q_{ij} a quick update. Boxes represent the available data: dark-shading for out-of-domain and the light-shading for in-domain.

Config./ Task	Prep.	Alignment	PT	LM
OLD-NEW	1K	3K	3K	3K
IN-OUT	1K	13K	13K	13K
3-TABLES	1K	3K	3K	3K
BATCH	1K	1,013K	1,013K	1,013K

Table 10: An example of the required effort of each configuration.

Here we assume that all data received in between slow updates is small and can be processed together. Preprocessing need not be repeated, but the other steps may perform better given more data. As seen in the table, both OLD-NEW and 3-TABLES require minimal processing. The difference between them is the way the previously-obtained data is stored; IN-OUT requires a more substantial amount of processing, and BATCH requires all the data to be processed from scratch.

So far, our discussion focused on phrase tables. Concerning the LM, in the Table 10 we assume one option, where the LMs configuration is equivalent to that of the phrase tables. Our experiments showed that – at least for the language pairs we assessed – the reordering model does not significantly affect translation performance; thus, we do not update it in any of the configurations. Tuning is discussed in Section 5.5.

5.4 Setting

Datasets

transLectures

We used the transLectures datasets for the language-pair English-French that were available at the end of Year 1. These consisted of a few thousand sentence pairs, that were produced through manual post-editing of the automatic transcription and then post-edition of the automatic translation of the transcriptions. The continuous text of the lectures was split into sentences based on long silences in the speech and with a maximal sentence-length constraint. This dataset and its production represent a typical scenario were in-domain spoken language data is scarce, hard to collect and slow to arrive.

- Training set: ∼4,000 English-French sentence pairs.
- Development set: 1,000 bi-sentences, used for tuning.
- Test set: 1,360 sentences-pairs.

WIT3

To confirm the validity of the results across datasets and language pairs, and to allow reproducing our results through a freely available resource, we used another spoken-language dataset, WIT3 [7]. WIT3 (Web Inventory of Transcribed and Translated Talks) is a parallel corpus created from transcription and translation of TED talks.[8] We used a different language pair, Italian to English, and 10-times as much training data as the TL dataset.

- Training set: 40,000 sentence-pairs from the Italian-English WIT3 corpus.[9]
- Development set: 1,000 bi-sentences of that corpus.
- Test set: 1,000 bi-sentences from the above corpus.

Europarl

For each language-pair we finally used, as part of the training set of most configurations, 1 million Europarl vs. 7 bi-sentences.

Experimental setup

Phrase-based SMT

Moses [13] was the translation system used for our experiments. When more than one phrase table was used, we used the *either* option, meaning that translation options are searched for in either table with no preference to one table over the other, while not expecting every translation option to be present in both tables.

Alignment

Some experiments assessed the use of incremental training and dynamic suffix arrays. For fair comparison, we used Incremental GIZA [16] in all our experiments.

Language Model

We trained 5-gram language models on the target side of the training set(s) using SRILM [24], with modified Kneser-Ney discounting [8].

Tuning

Model weights were tuned with batch MIRA [9].

Evaluation

We use Smooth (sentence-level) BLEU [5], and report the average score over the test set sentences. All our evaluations were performed on lower case, tokenized texts, using the standard Moses tools for preprocessing.

5.5 Experiments and results

In this section we present experiments conducted with the TL and WIT3 datasets, and their results.

[8] `http://www.ted.com`
[9] Downloaded from `https://wit3.fbk.eu`

Batch updates

We start by providing the results of "regular" batch updates, where the entire training set is used as a single corpus. The first row of each dataset in Table 11 shows the baseline, when no new data is used. This is the starting point of a system that was trained on a large amount of out-of-domain data; in the second row we show the result when 4K (TL) or 40K (WIT3) bi-sentences are used to update the phrase table (i.e. the translation model), but not the LM or the reordering table; the third row shows results of updating all three.

Dataset	Configuration	BLEU
transLectures	Baseline	23.9
	BATCH, PT only	27.9
	BATCH, complete	28.3
WIT3	Baseline	29.4
	BATCH, PT only	30.9
	BATCH, complete	30.7

Table 11: Results of batch updates.

Unsurprisingly, the addition of the new in-domain data to the phrase table greatly improves the translation quality; updating the LM and reordering tables adds a bit more on top of that for transLectures. As mentioned, initial experiments showed that reordering had insignificant impact on results, and effect may be mostly attributed to the LM update; we therefore assessed the performance of all following models without updating the reordering table.

Quick updates

We now evaluate the performance of quick update models. Table 12 shows the results of the three configurations where only the phrase table is being updated with the new data. That is, the language model and the reordering model are not updated at all. While using the same amount of data as for the batch updates in Table 11, and even with this partial model update, each of these configurations outperforms the batch update, over the two datasets. This result is consistent with prior work on domain adaptation, but the important aspect that we are concerned with is that this update is **much** faster. Instead of processing over a million sentence pairs, only 4,000 (TL) or 40,000 (WIT3) need to be handled.

Dataset	Configuration	BLEU
transLectures	OLD-NEW	29.4
	IN-OUT	29.7
	3-TABLES	30.2
WIT3	OLD-NEW	31.2
	IN-OUT	31.7
	3-TABLES	31.2

Table 12: Quick updates, where only phrase tables are updated.

Quick updates of the language model

Next, we evaluate the performance when the LM is also updated. We use multiple LMs, separated the same way as the phrase tables. This allows quick update of this model as well. Table 13 shows the results of this set of experiments.

Dataset	Configuration	BLEU
transLectures	OLD-NEW	31.2
	IN-OUT	31.8
	3-TABLES	31.6
WIT3	OLD-NEW	32.3
	IN-OUT	33.1
	3-TABLES	32.3

Table 13: Quick update results, with matching LM and phrase table configurations.

In all cases, results are improved relative to updating only the phrase-table (Table 12). Updating the LM was expected to help, yet here we experimentally see that even a quick LM update achieves significant improvements, and is useful for our goal. The best configuration is IN-OUT for both datasets. This is the slowest of the three configurations; hence, depending on the data size, the other options may also be considered, and in particular the 3-TABLES option.

We have seen that the LM quick update on top of the phrase table helps; we now wish to verify that updating the LM alone is not sufficient. Table 14 shows two such experiments on the WIT3 dataset. In the first, the target side of the WIT3 training corpus was added to the Europarl corpus to generate a single LM; in the second, the same WIT3 data was used to produce a separate LM. Note that the first among these is not a quick update per-se. Yet, LM generation is much faster than phrase table construction; if the performance is competitive, this can also be an option to consider.

As it turns out, training of a single LM with the additional data did not improve results relative to the baseline. Possibly, in-domain data (consisting of less than 4% or the training data in this case), is diluted in the entire set. More importantly, we see that the quicker update where the LMs are separated, is better. The performance is similar to the configuration where only the phrase table is updated but is inferior to all configurations where both models are updated.

Configuration	**BLEU**
Single LM	29.4
Separate LMs	31.4

Table 14: WIT3, updating only the language model.

Separating the LMs for batch training

Following the above results where LM separation helps, we assess this option with batch updates as well. Here we maintain a single phrase table, and separate only the LMs. This setting is still slow, yet somewhat quicker than a complete batch update since the previous LM need not be generated, just the new one. The more time-consuming steps of alignment and and phrase table construction are still necessary.

Dataset	Configuration	BLEU
TL	BATCH, single LM	28.3
	BATCH, separate LMs	31.6
WIT3	BATCH, single LM	30.7
	BATCH, separate LMs	32.6

Table 15: Comparison of batch configurations, with and without separating the LMs for in/out-of domain data. The single-LM BATCH configurations are the same ones shown in Table 11.

Table 15 shows that LM separation significantly improves results also when the PT is batch-trained, and while not considered quick, it is useful to separate the LMs between domains also in this case. The results are still inferior to those obtained by a complete in-out separation, and are just slightly better than other quick configurations in Table 13.

No-adaptation

So far our results included two types of datasets. We also wish to understand the effect of the different configurations when only a single domain is concerned. In this setting, IN-OUT and 3-TABLES are not relevant, only OLD-NEW is, with or without phrase table and LM separation. The TL data is too small for this experiment, and we use only WIT3, training a model with $30K$ sentence-pairs and updating it with additional $10K$. The results are shown in Table 16. The first row shows the baseline result before the $10K$ dataset is used, and the second shows the result where all data is trained together in a batch setting. The next two rows show quicker updates: the first – and the quickest – where both phrase table and LMs are separated between old and new data, and the second, where only the phrase tables are separated.

Configuration	BLEU
Baseline	28.2
BATCH	29.2
OLD-NEW	28.5
OLD-NEW, single LM	28.9

Table 16: WIT3 results, where only in-domain data is used.

Now that domain adaptation is no longer a factor, BATCH achieves the best result. Here, we can see the benefit of generating models using the entire data. Quick updates are not far behind, and are, well, quicker. In this setting, separating LMs of the same domain is not useful, and a better model is obtained when more data is used. Notice, though, that these scores are inferior to those obtained in previous experiments. Out-of-domain data is very useful, and as this is case, quick update methods should still be considered.

Tuning

In the experiments presented above, for consistency, tuning was conducted for each of the models with the actual data used for generating the model. Still, separate experiments show that tuning is not strictly required for every update. Tuning is likely necessary when a configuration is changing, either in terms of components, the data split between them, or in terms of the balance between the datasets. When these remain relatively fixed, and a small amount of data is added, tuning may be skipped. Two examples are shown in Table 17. In each, the first row shows the results of a model trained with the Europarl corpus and with partial TL data. The next two models (rows 2 & 3 for each experiment) use additional 1,000 TL bi-sentences and differ only in the tuning – while the first was re-tuned, the second was not, and used instead the tuned weights of the baseline model. We can see that by re-tuning we obtain a small gain in terms of performance; yet, we greatly lose in terms of time. In most cases, then, tuning can be skipped for intermediate updates, and reserved only for slow updates.

So far we have seen several options for model updates that can be applied very quickly. Using the Moses server, once an updated model has been generated, it can be loaded into memory practically instantaneously, replacing a previous instance of the server that was loaded with a

Setting	Configuration	BLEU
TL, 2K; IN-OUT	Baseline	27.76
	Re-tuned	28.45
	Not re-tuned	28.31
TL, 4K; OLD-NEW	Baseline	28.51
	Re-tuned	29.37
	Not re-tuned	29.19

Table 17: Tuning with all available data vs. using a model with the same configuration tuned with a smaller amount of data.

previous model. That is, as long as all large models are binarized. We can assume binarizing is done during slow updates, and that small models can be loaded quickly and fit into memory easily. With IN-OUT we run into the risk that in-domain data also becomes large; this is not an issue for the 3-TABLES configuration, where the processed data always remains small.

Incremental training and dynamic suffix arrays

We have extensively experimented with incremental GIZA, and with updates through the dynamic suffix array in Moses.[10]

Suffix arrays constitute an alternative to phrase tables, where the entire training data is maintained in memory rather than in a phrase table. Dynamic suffix arrays [16] further enable inserting or deleting training instances, thus updating the translation model without retraining [3]. Although very efficient in comparison to batch training, the process of incrementally updating a model with new data with these tools is not as fast as one would expect. Apart from preprocessing and alignment of the new data (which are required in any case), it requires updating the vocabulary and cooccurrence files, as well as the HMM probabilities, before the alignment can be performed. These statistic updates, which operate over the respective files of the entire data, need to be done independently of the size of the new data. It is therefore not efficient to run it per sentence, but rather per mini-batch. Once the new data has been aligned, inserting each bi-sentence into the suffix array is needed to have the translation system updated. Apparently, this is a time consuming process and cannot be considered real-time update. Creating a phrase table for the new data, and reloading another instance of the Moses server, is significantly faster.

We have run multiple comparative experiments with phrase tables vs. suffix arrays, and with combinations of them both, and observed a significant drop in results whenever the suffix array was used, with or without Incremental GIZA. A possible reason is the fact that the inverse translation probabilities were missing in this data structure; yet, an experiment we ran where the same features were removed from the phrase table as well, did not support this explanation. Moreover, when an update takes place, the translation server becomes unusable, maintaining the suffix array in memory takes up a large amount of memory and the updated model cannot be saved into disk, but needs to be reconstructed later. Further, updates to the LM are not supported, although this issue was addressed in [15]. All these make this data-structure currently difficult to use or rely on.[11]

The potential advantage of principled incremental training is obvious. Taking into account the previously accumulated data is expected to produce better statistics; doing so while maintaining the system live and constantly updated is a highly sought-after goal. Yet, aligning all

[10] We thank Abby Levenberg for his support at this part of the study.

[11] In summer 2013 a new implementation of the dynamic suffix array has been introduced in Moses, where all standard 5 features are computed. Some of the above issues may have been handled. To the best of our knowledge this is still work-in-progress and so far we have not experimented with it.

data, regardless of the domain, is not always beneficial. Thus, once such tools are stable and efficient, quick updates may be used in conjunction with incremental training and suffix arrays. For instance, out-of-domain data can be maintained in a phrase table, while in-domain data that needs updating is loaded into a suffix array. Preparations for alignment are longer, but the advantage in comparison to IN-OUT is that only alignment of the new data is necessary, but not phrase-table generation.

5.6 Quick-update approach: conclusions and recommendations

We have focused here on identifying simple configurations of phrase-based SMT systems that allow updating the underlying model quickly, when new training data becomes available. We have emphasized the applicability to domain adaptation, which is particularly relevant for spoken-data transcriptions contexts such as transLectures, where seed in-domain parallel resources are typically scarce. Still, the experiments show that this type of updates is suitable also for single-domain settings. Multiple configurations were assessed, some of which are based on proven methods from domain adaptation research, to highlight the preferred ones both in terms of translation quality and of processing speed. We described how quick updates can be integrated into the lifecycle of an operational SMT system, enabling efficiently maintaining translation quality while keeping the system up and up-to-date.

Our results show that quick updates are competitive with batch retraining on corpus concatenation, a strong baseline, while being orders of magnitude faster. We have seen that a complete separation of in- and out-of-domain data usually results with best translation quality; yet, this option may become slower over time. The 3-TABLES configuration we proposed solves this issue, albeit at the price of some drop in performance. A potential improvement for this configuration would be to reserve some, moderate size in-domain data for training together with the new data, benefiting from the potential improved alignment, while still keeping the update fast. Another interesting option to explore would be to use incremental GIZA to align the new data, but then to load only the new data into a separate phrase table.

Recommendations in the context of the transLectures workflow

The specific setting that will eventually be applied to transLectures depends, among other things, on the amount of post-edition feedback that is flowing in as well as the computational resources that are devoted for the update process, and will be determined experimentally, based on real data flows. Still, we provide below some general guidelines based on our experiments.

As mentioned above, quick updates can be performed, for instance, daily, while slow updates may be carried out once a week. Once the workflow is in place, the frequency of the updates can be easily adapted to the actual needs.

Quick updates

Following our experiments, and the time required to perform each task, we look separately at phrase tables and language models for quick updates:

Language models. Since training a language model is typically much faster than phrase table construction (and fast in actual time), we recommend using the IN-OUT configuration for language models. That is, training one LM for all out-of-domain data, and at every quick update, re-training a second LM with all transLectures data.

Phrase tables. Here we consider two cases:

1. All accumulated transLectures data is small, and can be processed quickly together. In this case, we recommend using the IN-OUT configuration, where all transLectures data is aligned and processed together to construct a single phrase table, in addition to the existing out-of-domain phrase table that need not be updated. As shown in our experiments, this is the configuration that yields the best results.

2. transLectures data has grown to the point that makes it computationally costly to process in a quick update. In such a case, we recommend using the 3-TABLES configuration. The out-of-domain table does not change in comparison to the previous case, but the TL data is now split between two phrase tables. The first is fixed with respect to quick updates, while the second is updated regularly. Assuming that post-edited data does not flow in very large daily quantities, we suggest including in the third table not only the data that was received since the previous batch update; rather, we reserve some of the older TL data, of an amount that can be processed fast, and training it together with the new data. This should make the statistics for this table more reliable, while not slowing down the process too much.

Slow updates

In the weekly (say) slow update, we can carry out any processes that require more time, including:

- Updating the out-of-domain table with additional data if any was received.
- Generating an updated reordering model.
- In the 3-TABLES setting: re-distributing the TL data between two in-domain sets (tables), such that most data goes to the first "fixed" one among them, and the rest to the constantly-updated one, and re-training them both.
- Binarizing all models.
- Re-tuning the model based on the entire received data.

Once these have been completed, we're ready for a new cycle of quick updates.

Lastly, we reiterate our previous recommendation to **not** use Moses' dynamic suffix arrays at this point. However, since this topic has recently received attention by Moses developers, we propose to be attentive with respect to the status of this component and consider including it in the workflow in the future. A potential advantage would be the ability to update the models in real-time.

References

[1] Peter F. Brown, Vincent J. Della Pietra, Stephen A. Della Pietra, and Robert L. Mercer. The mathematics of statistical machine translation: parameter estimation. *Computational Linguistics*, 19(2):263–311, 1993.

[2] P.F. Brown, J. Cocke, S.A. Della Pietra, V.J. Della Pietra, F. Jelinek, J.D. Lafferty, R.L. Mercer, and P.S. Roossin. A statistical approach to machine translation. *Computational Linguistics*, 16(2):79–85, 1990.

[3] Chris Callison-Burch and Colin Bannard. A compact data structure for searchable translation memories. In *Proc. of EAMT 10th Annual Conference (EAMT 2005)*, 2005.

[4] Olivier Cappé and Eric Moulines. On-line expectation–maximization algorithm for latent data models. *Journal of the Royal Statistical Society: Series B (Statistical Methodology)*, 71(3):593–613, 2009.

[5] Daniel Cer, Michel Galley, Daniel Jurafsky, and Christopher D. Manning. Phrasal: A statistical machine translation toolkit for exploring new model features. In *Proc. of the NAACL HLT Demonstration Session*, 2010.

[6] N. Cesa-Bianchi, G. Reverberi, and S. Szedmak. Online learning algorithms for computer-assisted translation. In *Deliverable D4.2, EU Project SMART*. 2008.

[7] Mauro Cettolo, Christian Girardi, and Marcello Federico. Wit^3: Web inventory of transcribed and translated talks. In *Proc. of EAMT 16th Annual Conference (EAMT 2012)*, pages 261–268, Trento, Italy, 2012.

[8] Stanley F. Chen and Joshua Goodman. An empirical study of smoothing techniques for language modeling. In *Proc. of the 34th annual meeting on Association for Computational Linguistics (ACL 1996)*, pages 310–318, 1996.

[9] Colin Cherry and George Foster. Batch tuning strategies for statistical machine translation. In *Proc. of the 2012 Conf. of the North American Chapter of the Association for Computational Linguistics: Human Language Technologies (NAACL-HLT 2012)*, pages 427–436, 2012.

[10] George Foster and Roland Kuhn. Mixture-model adaptation for SMT. In *Proc. of the Second Workshop on Statistical Machine Translation (StatMT 2007)*, pages 128–135, 2007.

[11] Qin Gao and Stephan Vogel. Parallel implementations of word alignment tool. In *Proc. of the ACL Software Engineering, Testing, and Quality Assurance Workshop (SETQA-NLP 2008)*, pages 49–57, 2008.

[12] Philipp Koehn. Europarl: A parallel corpus for statistical machine translation. In *Proc. of 10th Machine Translation Summit*, pages 79–86, 2005.

[13] Philipp Koehn, Hieu Hoang, Alexandra Birch, Chris Callison-Burch, Marcello Federico, Nicola Bertoldi, Brooke Cowan, Wade Shen, Christine Moran, Richard Zens, Chris Dyer, Ondrej Bojar, Alexandra Constantin, and Evan Herbst. Moses: Open source toolkit for statistical machine translation. In *Proc. of the 45th Annual Meeting of the ACL on Interactive Poster and Demonstration Sessions (ACL 2007)*, pages 177–180, 2007.

[14] Philipp Koehn and Josh Schroeder. Experiments in domain adaptation for statistical machine translation. In *Proc. of the Second Workshop on Statistical Machine Translation (StatMT 2007)*, pages 224–227, 2007.

[15] Abby Levenberg. *Stream-based Statistical Machine Translation*. PhD thesis, University of Edinburgh, 2011.

[16] Abby Levenberg, Chris Callison-Burch, and Miles Osborne. Stream-based translation models for statistical machine translation. In *Proc. of Human Language Technologies: The 2010 Annual Conference of the North American Chapter of the Association for Computational Linguistics (HLT 2010)*, pages 394–402, 2010.

[17] Percy Liang and Dan Klein. Online em for unsupervised models. In *Proc. of Human Language Technologies: The 2009 Annual Conference of the North American Chapter of the Association for Computational Linguistics (NAACL 2009)*, pages 611–619, 2009.

[18] Radford Neal and Geoffrey Hinton. A view of the EM algorithm that justifies incremental, sparse, and other variants. In Michael I. Jordan, editor, *Learning in Graphical Models*, pages 355–368. MIT Press, 1999.

[19] Franz Josef Och and Hermann Ney. Improved statistical alignment models. In *Proc. of the 38th Annual Meeting on Association for Computational Linguistics (ACL 2000)*, pages 440–447, 2000.

[20] Daniel Ortiz-Martínez, Ismael García-Varea, and Francisco Casacuberta. Online learning for interactive statistical machine translation. In *Human Language Technologies: The 2010 Annual Conference of the North American Chapter of the Association for Computational Linguistics (HLT 2010)*, pages 546–554, 2010.

[21] Kishore Papineni et al. BLEU: A Method for Automatic Evaluation of Machine Translation. In *Proc. of the 40th Annual Meeting on Association for Computational Linguistics (ACL 2002)*, pages 311–318, 2002.

[22] Alberto Sanchis, Alfons Juan, and Enrique Vidal. A word-based naïve bayes classifier for confidence estimation in speech recognition. *IEEE Transactions on Audio, Speech, and Language Processing*, 20(2):565–574, 2012.

[23] M. Snover et al. A Study of Translation Error Rate with Targeted Human Annotation. In *In Proc. of Association for Machine Translation in the Americas (AMTA 2006)*, pages 223–231, 2006.

[24] Andreas Stolcke. SRILM - an extensible language modeling toolkit. In *Proc. Int. Conf. on Spoken Language Processing (INTERSPEECH 2002)*, pages 257–286, 2002.

[25] N Ueffing and H Ney. Word-level confidence estimation for machine translation. *Computational Linguistics*, 33(1):9–40, 2007.

[26] Stephan Vogel, Hermann Ney, and Christoph Tillmann. Hmm-based word alignment in statistical translation. In *Proceedings of the 16th conference on Computational linguistics - Volume 2 (COLING 1996)*, pages 836–841, 1996.

[27] F. Wessel, R. Schlüter, K. Macherey, and H. Ney. Confidence measures for large vocabulary continuous speech recognition. *IEEE Transactions on Speech and Audio Processing*, 9(3):288–298, 2001.

[28] Joern Wuebker, Matthias Huck, Stephan Peitz, Malte Nuhn, Markus Freitag, Jan-Thorsten Peter, Saab Mansour, and Hermann Ney. Jane 2: Open source phrase-based and hierarchical statistical machine translation. In *Proc. of Int. Conf. on Computational Linguistics (CICLing 2012)*, pages 483–491, Mumbai, India, December 2012.