transLectures

Transcription and Translation of Video Lectures

D4.1.3: Final report on intelligent interaction

UPVLC, XEROX, JSI-K4A, RWTH and EML

Distribution: Public

transLectures
Transcription and Translation of Video Lectures

ICT Project 287755 Deliverable D4.1.3

October 31, 2014

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, 2014
Actual date of delivery	October 31, 2014
Date of last update	October 31, 2014
Deliverable number	D4.1.3
Deliverable title	Final report on intelligent interaction
Type	Report
Status & version	v1.0
Number of pages	38
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 Media Europe (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, 46022 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

Contents

1 Introduction

During M25-M36, progress made in WP4 **Intelligent Interaction** can be summarized as follows.

Task 4.1: Fast Constrained Search.

In the final year of the project, for the transcription task, the main new development by RWTH was a technique for directly constraining infixes during the recognition process, thereby allowing the search to be used in multi-pass recognition systems. Evaluation performed on VideoLectures.NET (English) with the current best ASR system confirmed earlier results observed at UPVLC on Polimedia (Spanish). Specifically, that correcting the 10% lowest-confidence words leads to large improvements in the recognition results. In addition, the final UPVLC's constrained-search algorithms were added to the open-source transcription toolkit *TLK*.

For the translation task, the outcome at month 24 was that the most effective form of constrained search was the provision of translations of unknown source words by adding new phrase pairs to the translation model. During this final reporting period, this model was implemented in the open-source translation toolkit *Jane*. It was also tested more thoroughly and confirmed in Task 4.3 (see below).

Task 4.2: Intelligent Interaction for Transcription.

The outcome at the end of the second year of the project was that the most useful form of supervision was at that performed at the level of single low-confidence words, with relative reductions in WER in the order of 50% when supervising less than 20% of all words. In the third year, work consisted mainly in the confirmation of these techniques based on enhanced Spanish ASR models, and in their application to the UPVLC's English transcription system as developed in WP3.

Task 4.3: Intelligent Interaction for Translation.

Experiments carried out during the second year showed that supervision of low-confidence sentences produced better results than supervision of low-confidence words, in exchange for the same supervision effort. In the third year of the project, a method for detecting and supervising low-confidence phrases (sequences of consecutive words) was developed, and experiments were repeated with the most current MT systems. While results were better than with words, they were still slightly inferior to those yielded by the supervision of whole sentences.

Experiments were carried out to combine constrained search and intelligent interaction, revealing that providing translations for OOV words significantly improved English-German and English-Slovenian BLEU scores with only a small user supervision effort.

Task 4.4: Incremental Training for Translation.

By month 24, XRCE had developed an approach allowing "quick-updates" of a translation system when newly human-translated data became available, avoiding the need to slowly retrain the whole system from scratch and retraining solely on the new translations at a much higher speed.

During the third year of the project, we compared different ways of performing incremental word-alignment (the most expensive training step in the quick-update method) and confirmed the advantage of Incremental Giza++. We developed a technique for incrementally training the lexical re-ordering component of Moses. Finally, we assessed the effect of different orderings of the training data and showed that the choice of ordering has a significant impact on the results.

2 Fast Constrained Search

2.1 Prefix-constrained search for transcription

During M25-M36, RWTH improved their implementation of constrained search. Infixes can now be used directly during the recognition process, skipping the cutting and joining steps. It also enables the search to be used in multi-pass recognition systems.

To make a final evaluation, the same experiments as performed up to M24 were repeated using the current best recognition system. For single-pass recognition, the results are shown in Table 1. In general, WER is one to two percent lower than in the experiments performed in the previous period. Again, it can be seen that user supervision of just 10% of the total words leads to considerably lower word error rates.

Finally, fast-constrained search was performed on RWTH's best multi-pass recognition system. As shown in Table 2, the results are quite similar to those of the single-pass system. The initial WER is now 5% lower with 20.1%. For random replacement the error rate is lowered by 1%, and by an additional 0.5% when constrained search is applied. By replacing the least confident words, the effect is even greater, leading to a WER of 15.6%, though constrained search can only lower this result by 0.3%. The third method, that of replacing the longest incorrect words, achieves an error rate of 12.8% but constrained search is unable to improve on this result at all. These experiments again confirm UPVLC's findings. Especially interesting is the fact that the improvements yielding by constrained search decreases as the number of corrected segments increases. In the case of replacing longest incorrect words, nearly all possible substitution errors are removed and no further improvement can be achieved via constrained search. The remaining WER consists mainly of insertion errors, deletion errors and errors corresponding to false transcription data.

In the end, it can be stated that the recognition result can be greatly enhanced by correcting the 10% least confident recognised words.

Table 1: Transcription quality in terms of WER calculated on the VideoLectures.NET evalset 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	25.9	24.6	23.8
Least Confident	25.9	20.7	20.1
Incorrect longest word	25.9	18.6	17.8

Table 2: Transcription quality in terms of WER calculated on the VideoLectures.NET evalset 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	Constrained Search
Random	20.1	19.1	18.4
Least Confident	20.1	15.6	15.3
Incorrect longest word	20.1	12.8	12.8

2.2 Summary of progress

Work performed on this task has been focused in the development of fast algorithms that satisfy user constraints in conventional search techniques for both transcription and translation. The following points summarize briefly how this main objective has been achieved throughout the project:

- Constrained search for transcription.
  - *Development of a generalized version of the well-know Viterbi algorithm that is able to deal with user constraints.* User constraints correspond to speech segments which have been supervised by users and they are used as conditions to be met in the recomputation of hypotheses. During the first year of the project, user constraints were defined to deal with user supervisions at word-level (replacement, deletion or insertion). In the second year, this implementation was extended to phrase-level (sequences of consecutive words) in order to evaluate different intelligent interaction strategies in Task 4.2 (word-level or phrase-level). Both implementations were empirically compared within the scenario of intelligent interaction for transcription, but only minor differences were found in performance. Despite this, slight improvements in WER (around half a point) are achieved when the constrained search is applied to the recomputation of the automatic transcriptions based on user-supervised segments. The improvement decreases as the number of supervised segments increases, since it is more complicated to correct errors when the transcriptions are more accurate. Both implementations have been added as features to the TLK toolkit (see D3.2.3 for more details about TLK).
  - *UPVLC and RWTH have developed their own systems for fast-constrained search.* By M12, UPVLC presented a system based on generalized constraints whereas RWTH introduced a prefix-constrained search based on the open source speech recognition system RASR [18] (details in D4.1.1). In the period ending in M24, the RWTH implementation was extended to a multiple prefix-constrained search, allowing prefixes not only at the beginning of a sentence, but also inside a sentence. With the UPVLC's system already evaluated on the poliMedia corpus, RWTH now repeated the same experiments for the VideoLectures.NET corpus. For the initial recognition, RWTH's current best English recognition system was used (described in D.3.1.2). In the final reporting period, implementation was improved to allow multi-pass recognition. Finally, experiments by RWTH confirmed the findings of the UPVLC regarding user supervision. We can state that supervising only 10% of the recognized words greatly enhances the outcome when using the right strategy; here, by choosing the least confident words.

- Constrained search for translation.
  - *Efficient algorithms for integrating several types of user constraints.* During the first period, we considered two types of user input to guide the translation process: either an unordered set of words (bag-of-words) or an ordered sequence of words. Here, user input is limited to the target language. The constrained search algorithm is a generalized version of the standard Viterbi search. User input can be considered either as a soft constraint, where the decoder is simply rewarded for producing one of the specified words, or as a hard constraint, where only hypotheses that fully match the constraints are possible. As the latter can lead to cases where the search fails and no translation is produced, we found soft constraints to be more practicable. Another difficulty is how to handle out-of-vocabulary (OOV) words, which we solved by introducing back-off phrases for OOVs. However, experiments revealed that, for

MT, this type of constrained search reaps only small benefits, since user constraints were often ignored by the decoder. As a consequence, in the second year a more involved type of interaction was considered, where the user specifies pairs of source and target sequences that are correct translations of each other. Here, we found that asking the user to solely provide translations of OOV words yields improvements of 0.4% BLEU and 2.4% TER over sequential annotation with a simulated user effort of 1%. The lower performance in terms of BLEU is due to the nature of this quality measure, which weights matches for 1-, 2-, 3- and 4-grams equally. It has the additional benefit that OOVs are often technical terms that are repeated several times during a single lecture. Thus, a single annotation may be used to improve the translation of a lecture in several places and the user can directly observe the effect of his or her work. The algorithms described above have been implemented in RWTH's open source translation toolkit *Jane*, which is described in more detail in D3.2.3.

3 Intelligent Interaction for Transcription

In the transLectures project, we aimed to prove that the process of automatically recognizing VideoLectures.NET and poliMedia could be improved by collaborative users. Given that user supervisions are limited and time-consuming, the transLectures project places special emphasis on intelligent interaction between user and 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

The intelligent interaction (II) approach evaluated in the M12-M36 period is that illustrated in Fig. 1. In the initial approach we proposed building an ASR system from scratch. However, we found that this was unrealistic. It is worth noting that competitive ASR systems have been developed in WP3 and might be adopted to carry out the intelligent transcription of video lectures. To address this, we proposed the current approach applied during the M12-M36 period.

Briefly, let us assume that a set of new video lectures recorded by different speakers needs to be transcribed. Figure 1 depicts this situation where there are 23 videos from 5 different speakers (this corresponds exactly to the distribution of videos in the poliMedia test set defined in Task 6.1). These videos are split into several blocks of similar duration (blocks in Fig. 1 are represented in columns). It should be noted that the organization of the videos into blocks responds to adaptation needs, since models are retrained after the supervision of each block is carried out. Also, video lectures from one speaker should be distributed sequentially along the blocks to ensure optimal performance of adaptation techniques.

Secondly, transcriptions ("CC" in the figure, abbreviated from "closed captions") for each video are automatically produced using the available ASR system (Fig. 1(a)). In this figure, the WER indicates the transcription quality that could be achieved without user interaction. WER is displayed by block (smaller font size) and overall (larger font size).

Thirdly, II is applied for the supervision of the automatic transcriptions produced in the previous step (Fig. 1(b)). Note that in the II approach, collaborative users devote a limited amount of effort to supervising a given percentage of words of the automatic transcription. User effort is optimised by ordering the speech segments selected for supervision from lower to higher reliability based on confidence measures (CMs). 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 (red line in Fig. 1(b)). The next step is to automatically re-transcribe the non-supervised blocks using these adapted (and hopefully improved) system models (Fig. 1(c)).

The process of partial user supervision, model adaptation and re-transcription is repeated block by block until all available user effort has been spent (Fig. 1(d) to Fig. 1(g)). As a final step, constrained search is applied using the resulting adapted models and the supervised segments to produce some last improvements to the final transcriptions (Fig. 1(h)). As a result, automatic transcriptions are improved by distributing user effort efficiently in the supervision of lower confidence transcription segments. Moreover, automatic transcriptions have also been improved using adapted models based on user corrections and applying constrained search as a final refinement.

The II approach has been comparatively evaluated against a baseline interaction approach called Batch Interaction (BI) (Fig. 2).

As in II, initial automatic transcriptions are produced using the best available ASR system (Fig. 2(a)). In BI, user effort is invested in full to supervise entire transcriptions from start to finish, or 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(b), by considering that the first block has been fully supervised by users. Once user effort is completely invested, the perfect transcriptions of the supervised block are then used to adapt the underlying models (Fig. 2(b)). Similarly to II, the resulting updated models are ultimately used to re-transcribe the remainder of videos, improving overall transcription quality (Fig. 2(c)).

3.2 Summary of progress

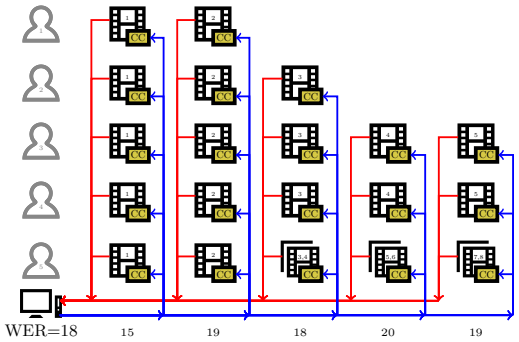
The II approach has been evaluated by simulating that the entire poliMedia and VideoLectures.NET test set has to be transcribed. The poliMedia test set has been split into 5 blocks. The main statistics on this distribution of videos into blocks are shown in Table 3. This table also shows the WER of the initial transcriptions produced by the UPVLC's best adapted Spanish ASR system developed in WP3 for each reporting period.

Table 3: Distribution of the poliMedia test set into blocks to evaluate intelligent interaction. For each block we show: time duration, running words and WER [%] (from M12 to M36). We also indicate totals.

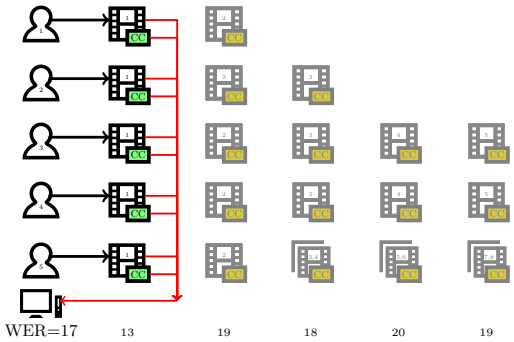
	Block					
	1	2	3	4	5	Total
Duration	51'01"	49'57"	39'28"	31'26"	35'14"	3h 27' 6"
Running words	7646	7471	5998	4232	4764	30111
WER (M12)	22.1	24.6	22.7	22.1	22.3	22.9
WER (M18)	21.5	23.4	22.6	21.1	21.2	22.1
WER (M24)	17.2	19.5	20.0	18.7	19.0	18.7
WER (M30)	13.5	14.8	16.2	16.1	14.8	14.9
WER (M36)	11.7	12.6	14.4	12.8	12.3	12.7

In the case of VideoLectures.NET, the video lectures have been organized into a single block, since there is only one video per speaker and so a block-based strategy for model adaptation is not applicable. The main statistics about this single block are shown in Table 4. The UPVLC's best adapted English ASR systems developed at M30 and M36 in WP3 have been used as initial systems.

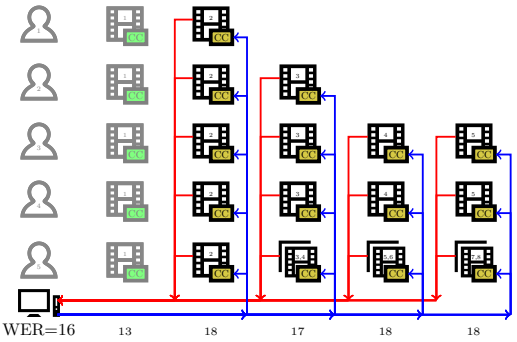
The II and BI approaches have been performed at two levels of supervision effort: 5% and 10%. These amounts represent the user effort that will be devoted to supervising output as



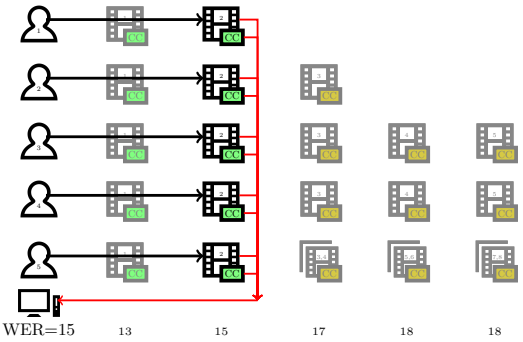
(a) First Transcription



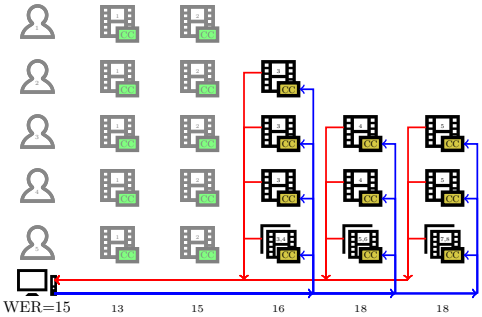
(b) First Interaction and Adaptation



(c) Second Transcription

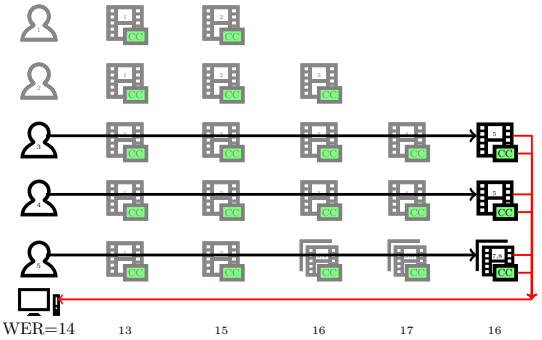


(d) Second Interaction and Adaptation

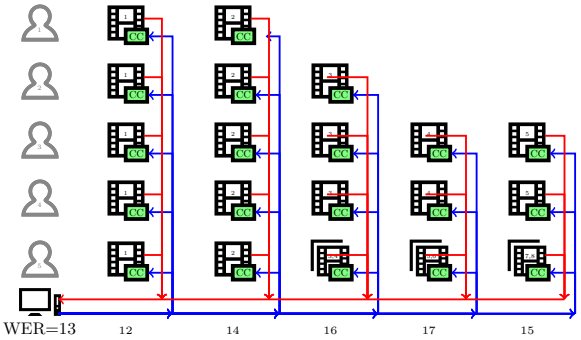


(e) Third Transcription

• • •

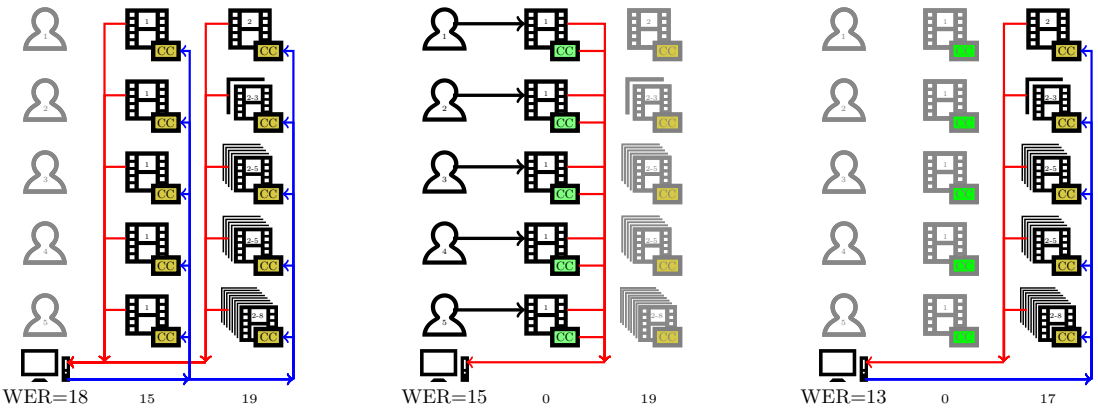


(g) Last Interaction and Adaptation



(h) Constrained Search

Figure 1: Intelligent interaction for transcription.



(a) First Transcription (b) Interaction and Adaptation (c) Final Transcription

Figure 2: Batch interaction for transcription.

Table 4: Time duration, running words and WER [%] (from M30 to M36) of the VideoLectures.NET single-block.

		WER	
Duration	Running words	M30	M36
3h 14' 4"	34441	24.8	22.9

a percentage of total words. In the case of II, the words to be supervised are distributed proportionately along the different blocks (i.e. 5% of words per block in the case of 5% of total effort).

Figure 3 depicts the progress of II techniques from M12 to M36 for poliMedia and from M30 to M36 for VideoLectures.NET. The plots on the left show the resulting WER for the transcriptions once the supervision process has been carried out following both the II and BI approaches. The initial WER of the transcriptions produced by the WP3 ASR systems is also plotted. The plots on the right show the relative reduction in WER over the WP3 ASR system performance achieved by both II and BI approaches.

As is shown in Fig. 3, II clearly outperforms BI at the same level of user effort in poliMedia and VideoLectures.NET. In both cases, II achieves relative reductions in WER that are approximately three times greater than in BI. The main reason for this superior performance is how good CMs are at locating misrecognized words. Note that following the BI approach, words are supervised in the order in which they appear. Making the assumption that errors are uniformly distributed in the transcriptions, the expected relative reduction in WER will be approximately equal to supervision effort, or slightly better due to the model adaptation and re-transcription carried out in a final stage. This is confirmed in Fig. 3, where relative reductions in WER are close to the user effort in the case of the BI approach. By contrast, the supervision of words from lowest to highest CM leads to higher relative reductions in WER than the level of user effort, since misrecognized words are successfully located. For instance, let us consider the WER of the initial transcriptions at M30 (15%) in poliMedia. Following the BI approach, one error would be amended each 6-7 supervised words. By contrast, three words would be amended using the II approach.

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. As with the

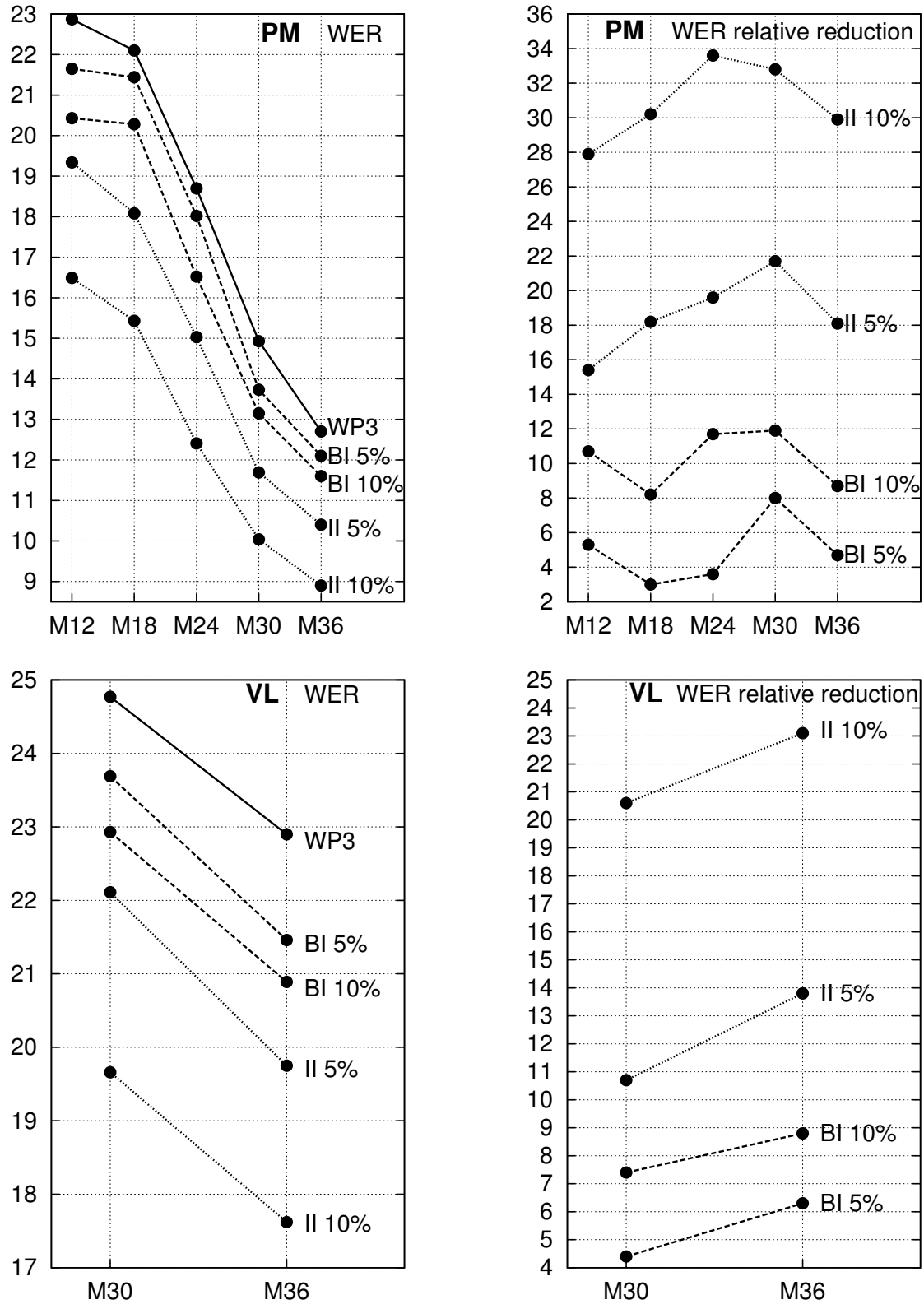


Figure 3: On the left, progress in Intelligent Interaction is given in terms of WER (poliMedia at top; VideoLectures.NET at bottom). Intelligent Interaction (II) is compared with Batch Interaction (BI) at two levels of supervision effort: 5% and 10%. "WP3" corresponds to the WER obtained by the UPVLC's best ASR systems developed in WP3. On the right, we see progress in terms of relative reduction in WER over the WP3 ASR system performance achieved for each approach.

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 this reason, 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

The intelligent interaction (II) approach that has been evaluated within the M12-M36 period is illustrated in Fig. 4. This approach was proposed in the M12-M24 period to better fit the initial approach into the transLectures context. From M12 until project end, the II approach has been evaluated on the scientific tests of transLectures and using the best adapted systems developed in WP3 as SMT engines.

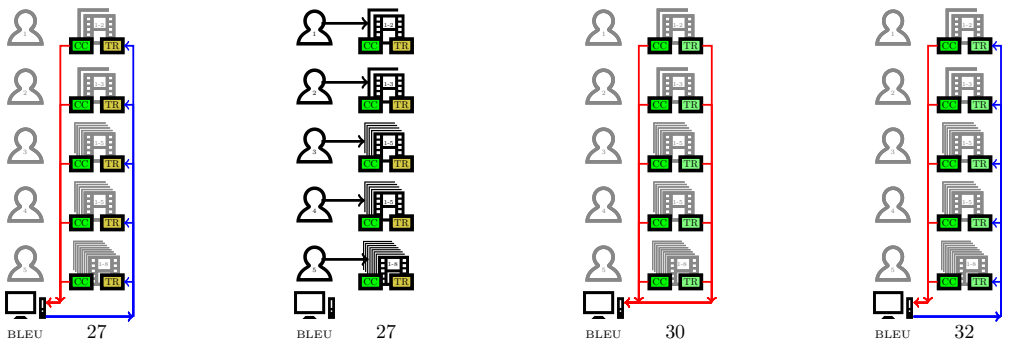
Briefly, let us assume that a set of new video lectures from different speakers have to be translated. Figure 4 depicts this situation where there are 23 videos from 5 different speakers. As the number of sentences in transLectures test sets is too small to adapt the system several times, video lectures have been organized in a single block.

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. 4(a)), translations (TR in the figure) are automatically produced for each video by translating the transcriptions using the best SMT system available for the language pair in question. In Fig. 4(a) BLEU [17] 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. 4(b)). 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. User effort is efficiently employed by supervising translation segments from lower to higher reliability based on confidence measures (CMs).

Once translation quality has been improved via user supervision, supervised segments of these translations are used to adapt SMT models (Fig. 4(c)). As a final step, non-supervised translations are automatically regenerated (and hopefully improved) using these adapted system models.



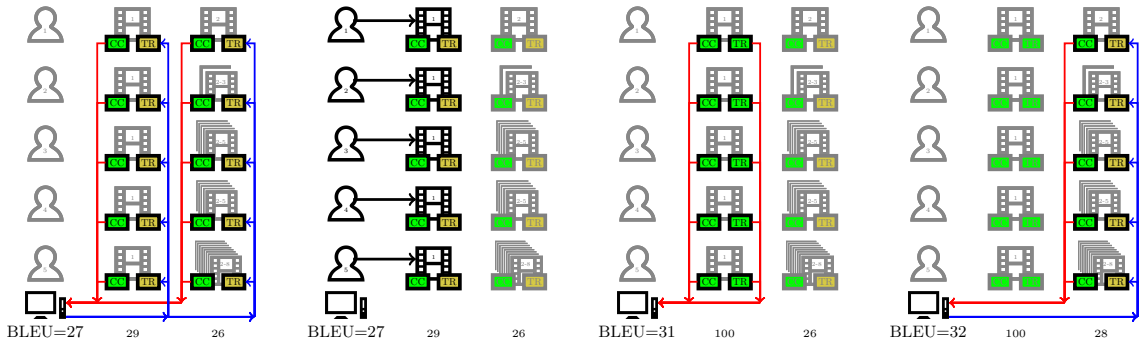
(a) First Translation (b) Interaction (c) Adaptation (d) Last Translation

Figure 4: Intelligent interaction for translation.

As in the case of II for transcription, II for translation has also been comparatively evaluated against batch interaction (BI). The BI approach is illustrated in Fig. 5. As we can observe, videos are organized into two blocks. 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 would correspond to the user effort.

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 (Fig. 5(a)). 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. 5(b)), automatic translations of the first block are fully supervised by users, producing perfect translations.

Once user effort has been completely invested, perfect translations of the supervised block are then used to adapt the SMT models (Fig. 5(c)). Finally, the adapted SMT system is used to translate the remainder of the videos improving translation quality (Fig. 5(d)).



(a) First Translation (b) Interaction (c) Adaptation (d) Last Translation

Figure 5: Batch interaction for translation.

4.2 Summary of progress

The II and BI approaches have been evaluated over the poliMedia and VideoLectures.NET test sets. It has been assumed that perfect transcriptions are available for all videos. The UPVLC's best (Spanish→English and English→Spanish) SMT systems developed in WP3 throughout the project has been used in the experiments. Both interaction approaches have been evaluated 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 three different interaction levels: sentence-level (IIS), sequences of consecutive words (phrase-level, IIP) and word-level (IIW). In all cases, supervision units (sentences, phrases or words) are supervised from lowest to highest reliability based on confidence measures (details about how the user interaction has been simulated in the experiments and confidence estimation can be found in the following sections).

Figure 6 depicts the progress of the task over the course of the project, from M18 for poliMedia and from M32 for VideoLectures.NET. For simplicity, only results using 5% and 10% of supervision efforts have been plotted (results for all supervision levels can be found in the following sections). Results for IIW are given from M24, since this kind of interaction was implemented within the M18-M24 period. In the same way, since IIP was implemented within the M24-M30 period, results are given from M30.

Experimental results show that II is not able to perform better than BI. II and BI achieve very similar performances when interaction is carried out at sentence-level, whereas lower levels

of interaction (phrases or words) do not produce improvements. Interaction at phrase-level largely outperforms word-level, but is slightly worse than sentence-level. Different factors have influenced this behaviour. On the one hand, as analyzed in D4.1.2, IIS has no potential to improve BI, since sentence-level is too coarse and the supervision of correct parts in a sentence consumes user effort without leading to any improvement. This explains why the random supervision of sentences (BI) yields very similar results to IIS. On the other hand, CMs at word-level are not good enough to rank the translated words effectively by correctness. This has a direct impact on the performance of II, since confidence measures in all cases are based on word-level. Comparatively, confidence measures at phrase-level present the best performance. This is because translation errors rarely correspond to isolated words and typically cover sequences of consecutive words. This better performance in confidence estimation at phrase-level explains why IIP largely outperforms IIW. Another remark is that user simulation in the experiments benefits both BI and IIS strategies, since the interaction is simulated at sentence-level. In both cases, user supervision produces the best possible output (i.e. reference sentences). In the case of word-level and phrase-level interaction strategies, the correction of non-sequential words has a much smaller impact on the overall BLEU score.

In the following sections, a more detailed description of the work developed in this M25-M36 period is provided, differentiating by type of interaction unit. Additionally, experiments on combining constrained search for translation and intelligent interaction focused on providing translations for out-of-vocabulary (OOV) words are also reported.

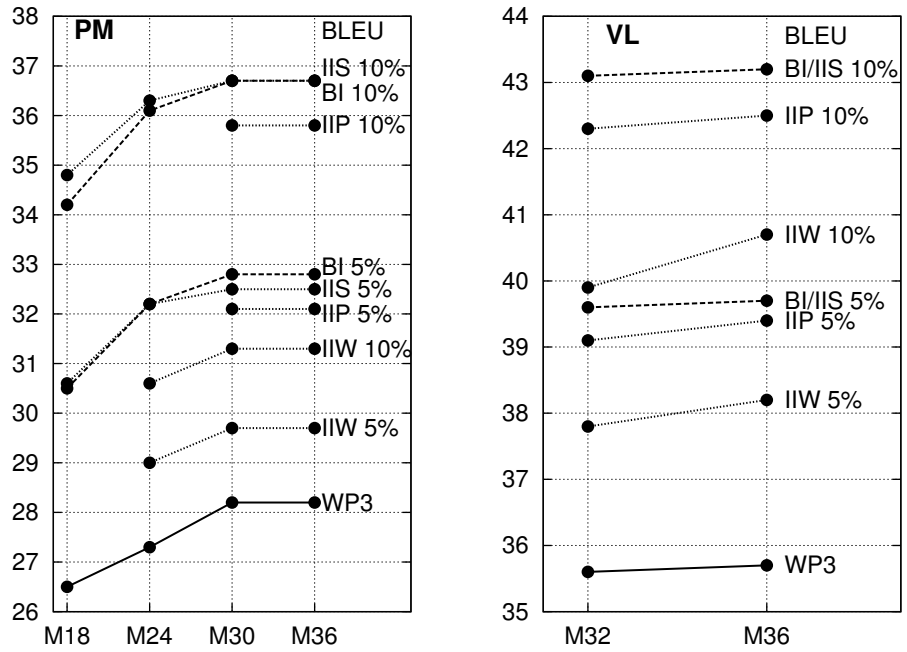


Figure 6: Progress in Intelligent Interaction given in terms of BLEU for poliMedia (left) and VideoLectures.NET (right). Intelligent Interaction (II) is compared with Batch Interaction (BI) at two levels of supervision (5% and 10%). II is performed using different kinds of interaction units: sentences (IIS), phrases (IIP) and words (IIW). "Best WP3" corresponds to the best WP3 UPVLC SMT systems.

4.3 Intelligent interaction at sentence-level

User simulation is a crucial part in the automatic assessment of the Intelligent Interaction approach. However, when the interaction is carried out at sentence-level this simulation is pretty straightforward. In this case, user simulation consists in just replacing the supervised sentence with its corresponding reference.

II at sentence-level is carried out by supervising sentences from lower to higher reliability based on CMs. As illustrated in Table 5, CMs have been computed based on sentence posterior probabilities. The calculation is performed using the N most probable translated sentences (i.e. N-best list). In this way, as is shown in Table 5, 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.

Table 5: Example of the 5 best translations of the Spanish sentence "Aquí tenéis un ejemplo". "SMT Scores" are the numerical 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

As in D4.1.2, an Oracle approach has been also tested based on the translation error rate (TER) [19] as an automatic measure of reliability. This allows us to assess the potential for improvement within a sentence-level interaction protocol. Table 6 shows the results achieved by the different interaction techniques in terms of BLEU.

Table 6: Results in terms of BLEU for the different sentence-based interaction strategies for translating the Spanish→English and the English→Spanish transLectures test sets at several supervision levels using different SMT systems. The size of the N-best lists was set to 2000.

	poliMedia				VideoLectures.NET							
	Es→En (M30)				En→Es (M32)				En→Es (M36)			
User effort	2%	5%	10%	20%	2%	5%	10%	20%	2%	5%	10%	20%
WP3	28.2				35.6				35.7			
BI	30.1	32.8	36.7	44.4	37.3	39.6	43.1	50.3	37.3	39.7	43.2	50.4
II	30.1	32.6	36.6	44.6	37.1	39.4	43.0	49.9	37.2	39.4	42.9	50.0
Oracle	30.2	32.7	36.9	45.1	37.1	39.3	42.8	49.9	37.3	39.4	42.8	50.0

Results confirm previous findings described in D4.1.2, showing that there is no room for improvement in the case of interaction at sentence-level. This conclusion is based on the minor differences between that achieved using the Oracle and other approaches. In some cases, the Oracle approach is worse than the others, showing that even TER is not useful for ranking hypotheses in order to produce improvements within an user interaction protocol. The main reason behind this similar behaviour is that the process does not seem to involve either very correct or very incorrect sentences. Thus, the supervision of correct parts of a sentence consumes user effort without leading to any improvement.

4.4 Intelligent interaction at word-level

User simulation at word-level is not as straightforward as in the case of sentence-level. In this case, as was presented in D4.1.2, word alignments are used to simulate which words would be replaced by the user to correct the erroneous translated word. Specifically, we use word alignments between the hypothesis and the source sentence, and between the source and the reference sentence, as depicted in Figure 7. In this example, the supervised word ("gets") 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, the supervision of "gets" would convert the translated sentence "to whom it gets?" into "to whom it are they filed before?".

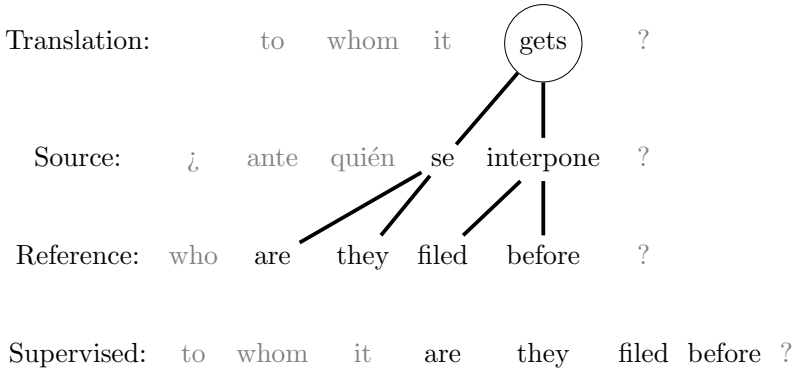


Figure 7: Example of user interaction at word-level.

II at word-level has been implemented using word-level CMs based on word posterior probabilities [21]. 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 illustrate each one of these methods by making use of the previous example in Table 5 to compute the CM of the "example". In the case of the Levenshtein position criterion, the 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 5 this would mean adding the sentence posterior probabilities of the first third of 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" in exactly the same position as in the 1-best. In Table 5 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 5 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 IBM Model 1 [1]. This confidence measure has been proven very useful for detecting mistranslated words [21]. Given the source sentence f_1^J and a translated word e, IBM1 confidence measure is defined as:

$$\text{IBM1}(e) \quad = \quad \max_{0 \leq j \leq J} p(e|f_j) \tag{1}$$

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 the sentence-level experiments, an Oracle approach has also been tested. The Oracle selects translated words that are not in the reference for supervision. It allows us to assess room for improvement within a word-level interaction protocol.

Table 7 shows the BLEU achieved after II is performed using each proposed method of confidence estimation and applying several levels of human effort (2%, 5%, 10% and 20%).

When computing BI, no re-ordering was allowed in the supervision of sentences. The purpose of this was to make word-level experiments more comparable. 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 7: Results in terms of BLEU for the different word-based interaction strategies for translating the Spanish→English and the English→Spanish transLectures test sets at several supervision levels using different SMT systems. The size of the N-best lists was set to 2000.

	poliMedia				VideoLectures.NET							
	Es→En (M30)				En→Es (M32)				En→Es (M36)			
User effort	2%	5%	10%	20%	2%	5%	10%	20%	2%	5%	10%	20%
WP3	28.2				35.6				35.7			
BI	28.8	29.7	31.2	34.5	36.2	37.2	38.7	42.7	36.3	37.4	39.0	42.2
II(Lev)	28.8	29.7	31.4	34.7	36.3	37.6	39.6	43.4	36.5	38.0	40.4	44.7
II(Fix)	28.8	29.6	30.8	33.5	36.1	37.1	38.4	41.6	36.3	37.5	39.0	42.3
II(Any)	28.9	30.0	31.5	34.8	36.4	37.8	39.9	44.2	36.6	38.2	40.7	45.1
II(Ibm)	28.8	29.6	30.9	33.1	36.1	37.6	39.8	42.7	36.2	37.7	40.0	43.2
II(Oracle)	29.2	31.1	34.5	40.9	37.2	39.7	43.4	51.1	37.3	39.9	43.4	51.4

4.5 Intelligent interaction at phrase-level

User simulation in phrase-level interaction is an extension of the method used in word-level interaction. By iteratively following the alignments between the hypothesis and the source, and the alignments between the source and the reference bidirectionally (forward and backward steps), we are able to obtain a consistent phrase for correction. Figure 8 illustrates the first iteration forward and backward steps. These forward and backward steps are repeated until the hypothesis phrase remains unchanged between iterations. When the iterative process is complete, the hypothesis phrase is replaced with the corresponding phrase from the reference.

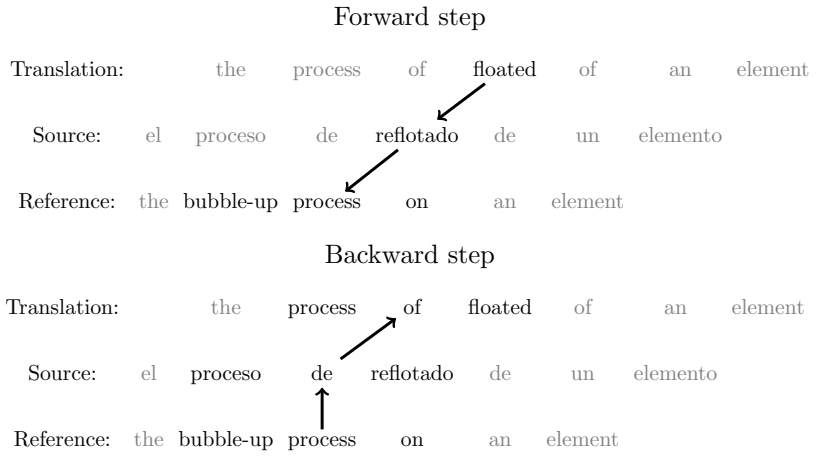


Figure 8: Example of user interaction at phrase-level.

The CMs used in these experiments are the same as those used in interaction at word-level. This means that CMs at word-level are used to supervise the words from lowest to highest reliability. However, supervision is extended to include the phrase that contains the word computed as explained in Fig. 8.

Table 8 shows the performance achieved using this kind of supervision. Important improvements in terms of BLEU are produced in comparison to word-level interaction, despite the CMs used being the same. It is observed that for this type of interaction, CMs based on IBM1 model work much better than any other CM. This is because the detection of mistranslated phrases is better using CMs based on the IBM1 model. In this case, the confidence score given to an incorrect word in an incorrect phrase will be lower than the confidence score assigned to a word that is in a phrase where all other words are correct. Note that in the former case, it will be harder to align properly and the IBM1 model score will be lower for the words in that phrase. This peculiarity also explains why CMs based on the IBM1 model are able to outperform Oracle supervision in some cases.

Table 8: Results in terms of BLEU for the different phrase-based interaction strategies for translating the Spanish→English and the English→Spanish transLectures test sets at several supervision levels using different SMT systems. The size of the N-best lists was set to 2000.

	poliMedia				VideoLectures.NET							
	Es→En (M30)				En→Es (M32)				En→Es (M36)			
User effort	2%	5%	10%	20%	2%	5%	10%	20%	2%	5%	10%	20%
WP3	28.2				35.6				35.7			
BI	29.0	30.2	32.4	36.7	36.4	37.4	39.1	42.5	36.6	37.6	39.4	43.5
II(Lev)	29.3	31.0	33.9	39.2	36.5	38.1	40.7	45.6	36.8	38.7	41.5	47.0
II(Fix)	29.1	30.5	33.1	37.6	36.3	37.6	39.7	43.3	36.6	38.1	40.1	44.1
II(Any)	29.5	31.5	34.4	40.0	36.6	38.3	40.9	46.2	36.8	38.8	42.0	47.5
II(Ibm)	29.9	32.4	35.6	40.1	37.0	39.1	42.3	46.8	37.2	39.4	42.5	47.8
II(Oracle)	29.5	32.0	36.2	44.2	37.3	40.0	44.0	52.4	37.3	40.1	43.9	52.6

4.6 Intelligent interaction on out-of-vocabulary words

As was described in Sec. 2.2 of D4.1.2, it can be beneficial to focus the user effort on providing translations for out-of-vocabulary (OOV) words, which are often technical terms. These usually appear multiple times in a single talk and can therefore be used to improve its translations throughout an entire video. We ran experiments on the English→German and English→Slovenian to investigate its benefits more thoroughly. Table 9 shows statistics on the OOV rates of the test sets. The English→German test set contains 312 OOV words, which corresponds to 0.9% of the total running words. Counting each unique word only once, this number goes down to 148 words, or 0.4%. For English→Slovenian the number of OOV words is 234 and 114, respectively, corresponding to 0.6% and 0.3% of the total number of source words.

Table 9: Number of OOV words on the source side of the English→German and English→Slovenian test sets.

	English→German	**English→Slovenian**
sentences	1360	1360
running words	36315	36133
running OOV words	312	234
unique OOV words	148	114

In our simulated experiments, we simulate a user who provides one translation for each OOV word, which corresponds to a total effort of 0.4% of the source words for English→German and 0.3% for English→Slovenian. As a baseline we compare it with sequential annotation of the

same number of words. Table 10 shows results in BLEU and TER. Sequential annotation yields small improvements of 0.3% BLEU and 0.2% TER on both tasks. Replacing all OOV words by their translation provided by the simulated user improves the result by 0.5% BLEU and 1.4% TER for English→German, but only by 0.2% TER for English→Slovenian. Furthermore, adding the provided translations to the phrase table and retranslating (constrained search) yields an additional improvement of 0.1% BLEU and 0.2% TER for English→German and 0.3% BLEU and 0.2% TER for English→Slovenian.

Table 10: Experimental results for intelligent interaction by annotating OOV words on the English→German and English→Slovenian translation tasks. In both cases, human effort was simulated by providing the most frequent translation for each out-of-vocabulary word.

system	**English→German**		**English→Slovenian**	
	BLEU[%]	TER[%]	BLEU[%]	TER[%]
baseline	20.6	63.6	19.8	62.8
sequential annotation	20.9	63.4	20.1	62.6
OOV annotation	21.4	62.0	20.1	62.4
+ constrained search	21.5	61.8	20.4	62.2

5 Incremental Training for Translation

Statistical machine translation systems rely on the availability of large parallel corpora, in particular of the target domain. Such corpora are not always available at the initial stage of the SMT model training, but are sometimes obtained during the lifetime of the system. More parallel data, especially in-domain, may become available, for instance, as users of the system *post-edit* the automatic translations. The source texts and their corrected translations then become new parallel corpora with which the system can be updated. It is then desirable to incorporate the new data into the SMT model as soon as possible. This is particularly a concern for interactive systems, where one wishes to reflect the corrections immediately to avoid repeating translation errors that have already been corrected. The straightforward way to incorporate new data into an SMT model is to retrain the model, i.e. to use all the data accumulated until that point and create the model all over again. However, such retraining may be a lengthy and computationally expensive process, leading to long lags between system updates. An alternative approach is to incrementally train the SMT model instead of retraining it, i.e. to update the previous model with the new data instead of executing a full re-train.

In this Section we describe the experiments conducted at XRCE to address this task in the third year of the project. In Section 5.1 we briefly describe the *quick updates* approach we developed in earlier stages of the project, and which we followed in the third year. In Section 5.2 we present an experiment for assessing the alignment component we use in our experiments, and in Section 5.3 we present a novel method to incrementally update the SMT reordering model. Lastly, in Section 5.4 we propose and explore the idea of ordering the training data for multi-cycle quick updates of the SMT model.

5.1 Quick updates of SMT models

For the task of incrementally training the SMT model, XRCE has focused on an approach consisting in short-cycle quick-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 were evaluated and were found to be proving a competitive and practical solution for interim updates of the model. 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. This approached, developed up to Y2, is described in detail in D4.1.2 and was published in [14]. We continued to work within this framework in Y3.

5.2 Alignment for Incremental Training

Word alignment is a critical step in the creation of phrase-based SMT models, and a prerequisite of many of the following steps of the model creation. Yet, this is by far the slowest process of the model training, which constitutes a primary motivation for using incremental training. In our work on Task 4.4 we have used Incremental GIZA++ [12] as our alignment tool. Briefly, Incremental GIZA allows updating the word alignment and translation probabilities with new data without retraining the model, by interpolating the counts of the old and the new data. It thus constitutes an important tool for incremental model updates within Moses. Based on initial experiments in Y2, we have concluded that the performance of Incremental GIZA in terms of translation quality is competitive to that of the default alignment tool used in Moses, GIZA++ [16]. In this section we provide actual results of comparing several options for aligning parallel data, showing that Incremental GIZA is a viable option to choose for the task of incremental training for translation.

Evaluating alignment

Alignment is an intermediate step in building SMT models. As a result and as the common practice, we do not assess alignment performance directly, but rather consider the translation performance, measured with BLEU, as a means to compare alignment alternatives, where the alignment tool is the only parameter of the experiment that is being changed. In the setting of incremental training, we assume that we have trained an MT model with initial data, and then, whenever more parallel data is becoming available, we update the MT model using it. That is, we engage in multiple update cycles. Our experiments simulate this scenario with an initial model created at step 0, followed by 10 update cycles.

The new data used in each update comes in the form of bi-sentences. To be able to include it in the model we need to word-align each source sentence to its corresponding target sentence. After each iteration, a new phrase table, a new reordering table and a new language models are generated from all the data. Due to the small side of the datasets, we did not use the quick update approach in these experiments.

We have performed the evaluation using the following alignment options:

- **MGIZA++** [6]: This is a parallel, and therefore faster, implementation of GIZA. Using it, we add the new training data to the old one and re-align **all** the corpus together.
- **Force-alignment** [5]:[1] Here, the new data is aligned based on the initial alignment model, without any updates to it.
- **Incremental GIZA++** [12]: Updating the alignment model with the new data. This implementation is using IBM Model 1 with HMM, in comparison to IBM Model 4 that is used by default by GIZA.

[1] http://www.kyloo.net/software/doku.php/mgiza:forcealignment

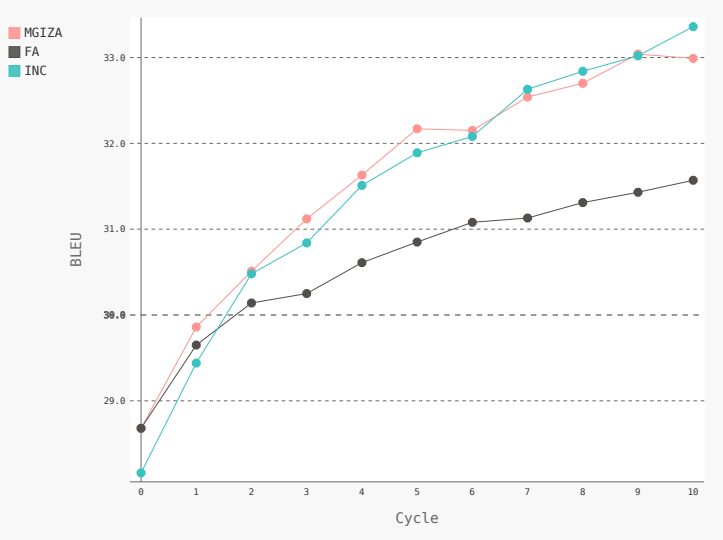


Figure 9: BLEU scores of an SMT system trained with additional data, over 10 cycles, using the different alignment options. *MGIZA* is the parallel implementation of GIZA, FA denotes force-alignment and INC denotes incremental GIZA.

Evaluation results

For our comparative experiment we used the IWSLT 2013 Evaluation Campaign data, of the English-French MT track.[2] The initial model was trained with 10,000 WIT3 [2] sentence-pairs; we use 50,000 additional ones to train updated models. The additional data is split into 10 parts of 5,000 bi-sentences, each added to the data used in the prior cycle to generate an updated model. Moses is used as the phrase-based SMT system,[3] with a configuration comprising of a single phrase table, a single language model and a single reordering model, all of which are updated after each cycle. 5-gram language models are trained over the target-side of the training data, using SRILM [20] with modified Kneser-Ney discounting [3]. Mean Error Rate Training (MERT) [15] is used for tuning the initial model using the development set of the above mentioned campaign, consisting of 887 sentence-pairs, and optimizing towards BLEU. The models are evaluated with BLEU over the campaign's test set of 1,664 bi-sentence. All datasets were tokenized, lowercased and cleaned using the standard Moses tools. None of the models is re-tuned after each cycle, and all methods use the tuning of the initial model.

Figure 9 shows the results of these experiments. It is clear from the figure that force-alignment, which does not update the model, is the least favorable option. On the other hand, Incremental GIZA performs as well as the complete batch update with MGIZA, but enables updating the model faster. Note that the initial model trained with Incremental GIZA obtains inferior performance to the one trained with MGIZA. Nevertheless, once more data is introduced, the model catches up with the batch-trained model. In sum, the results show that beyond the fact that Incremental GIZA allows performing incremental updates from a computational point of view, it is also a competitive option with regards to translation performance.

5.3 Incrementally Updating the Reordering Model

This part of XRCE's work, done in Y3 and published in [13], is concerned with incrementally updating the *reordering model* (RM), a component in phrase-based machine translation that

[2]Downloaded from https://wit3.fbk.eu/mt.php?release=2013-01.
[3]We used the version released on 14/3/2014.

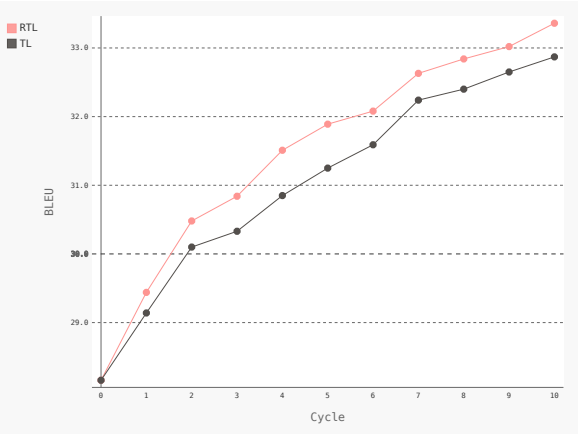


Figure 10: BLEU scores of an English to French SMT system trained with additional data, over 10 cycles, with and without updating the reordering model. R, T and L denote the models that have been updated – Reordering, Translation and Language models. The exact setting of this experiment, as well as additional details, are provided in Section 5.3.4.

models phrase order changes between the source and the target languages, and for which incremental training has not been proposed so far. First, we show that updating the reordering model is helpful for improving translation quality. Second, we present an algorithm for updating the reordering model within the popular Moses SMT system. Our method produces the exact same model as when training the model from scratch, but doing so much faster.

5.3.1 Reordering model incremental update: Introduction

Typical phrase-based SMT models use a log-linear combination of various features that mostly represent three sub-models: a *translation model* (TM), responsible for the selection of a target phrase for each source phrase, a *language model* (LM), addressing target language fluency, and a *reordering model* (RM). The reordering model is required since different languages exercise different syntactic ordering. For instance, adjectives in English precede the noun, while they typically follow the noun in French (*the blue sky* vs. *le ciel bleu*); in Modern Standard Arabic the verb precedes the subject, and in Japanese the verb comes last. As a result, source language phrases cannot be translated and placed in the same order in the generated translation in the target language, but phrase movements have to be considered. This is the role of the reordering model. Estimating the exact distance of movement for each phrase is too sparse; therefore, instead, the *lexicalized reordering model* [9] estimates phrase movements using only a few reordering types, such as a *monotonous* order, where the order is preserved, or a *swap*, when the order of two consecutive source phrases is inverted when their translations are placed in the target side.

Most research on incremental training for SMT addresses parallel corpus alignment, the slowest step of the model training and a prerequisite of many of the following steps, including the reordering model generation. Currently, keeping the reordering model up-to-date requires creating the model from the start. Refraining from updating this model is expected to yield inferior translation performance, since bi-phrases that are found in the phrase table but are absent from the reordering model receive a default score when selected. An example is shown in Figure 10, comparing results with and without an updated reordering model. While not as important a component as the TM or the LM (see further results in Section 5.3.4), updating the RM does improve translation. In contrast to the translation and language models, currently Moses supports a single reordering model. Hence, while it is possible to quickly create small TMs and LMs, as we do within the quick update approach, this is not possible for the reordering

model. Incremental updates of the RM have not been addressed yet and the only option currently available is to generate the reordering model from start, which might be a lengthy process. We therefore seek to allow quick incremental updates of the RM within Moses [11].

5.3.2 The Moses reordering model

Reordering probability estimation

The reordering model estimates the probability of phrase movements between the source and the target. To deal with sparsity, movement is measured in the lexicalized reordering model in terms of *orientation* types, rather than exact move distance. The default orientations used in Moses are listed below, and are referred to as *msd* [9]:

- *mono* (*monotonous*) – the preceding target phrase is aligned to the preceding source phrase.
- *swap*: the preceding target phrase is aligned to the following source phrase.
- *discontinued* (also called *other*): the phrases did not occur consecutively, but other phrases were inserted between them.

Formally, the probability of each of the above orientation types, o, for a source phrase f and a target phrase e is denoted $p(o|f,e)$. Counting the orientation instances of each phrase pair from the word alignments, in each direction, maximum likelihood is used to estimate this probability:

$$\hat{p}(o|f,e) = \frac{count(o,f,e)}{\sum_{o'} count(o',f,e)} = \frac{count(o,f,e)}{count(f,e)} \tag{2}$$

The estimation can be smoothed by additive (Laplace) smoothing with a factor σ:

$$\hat{p}(o|f,e) = \frac{\sigma + count(o,f,e)}{\sum_{o'} \sigma + count(f,e)} \tag{3}$$

Data structures

Extracted phrases

During the training of a phrase-based Moses model, phrase pairs are extracted from the word-aligned parallel data and used for training both the TM and the RM. Within the phrase extraction step, three files containing the list of phrase pairs are created. Two of them consist of the word alignments within the phrases, one in each direction (source-to-target and target-to-source); the third, the *reordering file*,[4] shows the orientation of each occurrence of the phrase pair, in either direction. Phrase pairs are alphabetically ordered in these files, and repeat if more than one instance of the phrase pair is encountered.

Figure 11 shows a few lines from a reordering file, of an English to French model, built with the *msd* (monotonous-swap-discontinued) orientations [10][5] Each line in the reordering file contains three parts, separated by '|||': source phrase, target phrase, and 2 indicators of the orientation in which this instance was found, when extracting the phrases from source-to-target and from target-to-source alignments.

[4] Not to be confused with the reordering *table*.

[5] More precisely, this is the *msd-bidirectional-fe* model, also referred to as *wbe-msd-bidirectional-fe-allff*.

```
but of course ||| mais bien sûr ||| mono mono
but of course ||| mais bien sûr ||| mono other
but of course ||| mais bien sûr ||| mono other
...
confusion between the ||| confusion entre le ||| other other
confusion between the ||| confusion parmi les ||| other mono
...
emerging ||| naissante ||| mono mono
emerging ||| naissante ||| other mono
emerging ||| naissante ||| other mono
emerging ||| naissante ||| other other
emerging ||| naissante ||| swap other
emerging ||| naissante ||| swap other
emerging ||| naissante ||| swap other
```

Figure 11: Sample lines from a Moses reordering file with *msd* orientations.

Reordering table

The *reordering table* (RT), created from the reordering file, is the data structure representing the reordering model. It contains probability estimations for each orientation of a phrase pair in either direction. In contrast to the reordering file, in the RT, each phrase pair appears only once. Figure 12 displays a few lines from a reordering table. In Section 5.3.3 we show how these estimations are computed.

```
but of course ||| mais bien sûr ||| 0.78 0.11 0.11 0.33 0.11 0.56
...
confusion between the ||| confusion entre le ||| 0.20 0.20 0.60 0.20 0.20 0.60
confusion between the ||| confusion parmi les ||| 0.20 0.20 0.60 0.60 0.20 0.20
...
emerging ||| naissante ||| 0.18 0.41 0.41 0.41 0.06 0.53
```

Figure 12: Sample lines from a Moses reordering table generated for the *msd* orientations, with 6 feature scores for each phrase pair. The scores are probability estimations, summing to 1 for each direction. For easier display, we round the scores to 2 places after the decimal point.

Reordering model generation

Several steps must be performed before a Moses RM can be trained. The necessary steps on which the model generation depends on are listed below.

1. Corpus preparation: tokenization, lowercasing and any other preprocessing.
2. Corpus alignment in both directions, source-to-target and target-to-source.
3. Bidirectional phrase extraction.
4. Creation of the reordering file.

Note that some steps are necessary for other purposes. For instance, Step 1 is necessary for all subsequent steps, including LM training, and Steps 2 and 3 are also necessary for training the TM. In practice, the creation of the reordering file (Step 4) is done within the phrase extraction step.

From the reordering file, the reordering table is created by counting the number of occurrences of each orientation in each direction and normalizing by the total number of occurrences of the phrase pair, as in Equation 3. In contrast to the reordering file, the reordering table contains only unique phrase pairs. To get an intuition of the involved sizes, a reordering file created from 500,000 lines of the tokenized, lowercased Europarl corpus [8] contains approximately 57M lines of non-unique phrase pairs, and the reordering table contains 33M pairs (58%); the figures for the complete Europarl corpus (1.96M lines after cleaning) are 219M for the reordering file in comparison to 107M lines for the RT (49%).[6]

[6]The more data we use, especially of the same domain, the fewer new phrase pairs we expect to see; since the RT, but not the reordering file, contains only unique phrase pairs, the ratio of their sizes is expected to decrease

```
but of course ||| mais bien sûr ||| 0.78 0.11 0.11 0.33 0.11 0.56 3
...
confusion between the ||| confusion entre le ||| 0.20 0.20 0.60 0.20 0.20 0.60 1
confusion between the ||| confusion parmi les ||| 0.20 0.20 0.60 0.60 0.20 0.20 1
...
emerging ||| naissante ||| 0.18 0.41 0.41 0.41 0.06 0.53 7
```

Figure 13: Sample lines from a reordering table with counts.

5.3.3 Merging reordering tables

We now present a simple algorithm for a reordering model update via the merge of two reordering tables. This update option requires keeping track of the number of occurrences of each phrase pair in the reordering table. We first present the format and technical details of this extension of the reordering table, and then provide the details of the suggested merge itself.

We denote the old, new or merged (updated) data with $\mathcal{A}$, $\mathcal{B}$ or $\mathcal{AB}$, respectively. For simplicity, we always refer below to the old data as $\mathcal{A}$ and to the new data as $\mathcal{B}$ without cycle indexes.[7]

We can assume that the data that was already used to train the current model (the older data) is significantly larger than the training data which we use for a single update (the newer data). This would typically be the case, for instance, with training data that is based on human feedback. As we proceed with subsequent update cycles, $\mathcal{A}$ keeps growing, while the size of $\mathcal{B}$ does not depend on prior cycles.

Reordering table with counts

To enable updating the table without generating it from scratch we must keep track of the number of occurrences of each phrase pair. To do it without making changes to Moses code, we add the total count of a phrase pair as an additional value following the feature scores in the reordering table. Figure 13 shows several lines of the reordering table shown earlier, now including counts.

Below is a demonstration of calculating the orientations scores in Figure 13 in the source-to-target direction, using Equation 3. In the equations below, $S(\cdot)$ is a scoring function and $C(\cdot)$ is a count function, using counts from the reordering file; f is *'emerging'* and e is *'naissante'* from Figure 13, which occur totally 7 times, out of which, the *mono* orientation occurs once in this direction, and each of *swap* and *other* occur 3 times. Each score is the result of smoothing the counts with a σ factor of 0.5 to avoid 0 probabilities. While demonstrated on the *msd* model, there is nothing that prevents applying the same approach to a different set of orientations.

$$S(mono|f,e) = \frac{\sigma + C(mono,f,e)}{3\sigma + C(f,e)} = \frac{0.5+1}{1.5+7} = 0.18 \tag{4}$$

and

$$S(swap|f,e) = \frac{\sigma + C(swap,f,e)}{3\sigma + C(f,e)} = \frac{0.5+3}{1.5+7} = 0.41 \tag{5}$$

Hence, recovering from the score the count of a specific orientation (e.g. *mono*) for a given phrase pair:

$$C(mono,f,e) \;=\; S(mono|f,e) \times (3\sigma + C(f,e)) - \sigma \;=\; 0.18 \times (1.5+7) - 0.5 \;=\; 1 \tag{6}$$

with more data.

[7] Denoting the initial "old" training data as $\mathcal{A}_0$ and the first new data as $\mathcal{B}_1$, $\mathcal{A}_i = \mathcal{A}_{i-1} \cup \mathcal{B}_i$, where $i = 1, 2, \ldots$ and '$\cup$' denotes the concatenation of the two training datasets.

```
LexicalReordering name=LexicalReordering0 num-features=7
type=wbe-msd-bidirectional-fe-allff input-factor=0
output-factor=0

LexicalReordering0= 0.0857977 0.0655027 0.0486593 0.115916 -0.0182552 0.0526204 0
```

Figure 14: An example Moses ini file with required changes to support RT counts.

To support RT with counts, the configuration (*ini*) file is adjusted to include 7 features instead of 6 (the number of features in the *msd* model), and its weight is set to 0. Figure 14 shows the relevant lines from a tuned configuration file, updated to support counts.

Merging RTs

Algorithm 1 presents the pseudo code of merging two reordering tables with counts, $R_{\mathcal{A}}$ and $R_{\mathcal{B}}$, into a single one, $R_{\mathcal{AB}}$. The procedure is as follows: We read the reordering tables in parallel, one line at a time, and compare the phrase pair in the old table with the one in the new one. The comparison is alphabetical, using a string made of the source phrase, the delimiter and the target phrase. When the two lines refer to different phrase pairs, we write into the merged table, R_{AB}, the one that alphabetically precedes the other, and read the next line from that table. If they refer to the same phrase pair we merge the lines into a single one, which we write into $R_{\mathcal{AB}}$, and advance in both tables. When one table has been read completely, we write the remainder of the other one into $R_{\mathcal{AB}}$.

Merging two lines into a single one (MERGE_LINES in Algorithm 1) consists of the following steps:

1. Convert the feature scores in each line into counts, as in Equation 6.
2. Sum up the counts for each orientation, as well as the total count.
3. Convert the updated counts of the orientations into scores, as in Equations 4 and 5.

The complexity of this algorithm is linear in the length of the tables. In terms of memory usage, neither table is fully loaded into memory. Instead, at any given time a single line from each table is read.

5.3.4 Evaluation

We evaluate updating the reordering model from two aspects: (i) translation performance and (ii) run-time. Specifically, we first show that updating this model helps improving translation, as reflected in the BLEU score [17]; then we show that the incremental update is faster than the complete one.

Our experiments use the same setting as in Section 5.2. Briefly, the initial model is trained with $10K$ WIT3 parallel sentences and the 10 subsequent models are generated by adding $5K$ bi-sentences at each cycle. In all our experiments, we use Incremental GIZA that allows updating the alignment and translation models without aligning all the training data at every cycle. With Incremental GIZA, the alignment of the parallel data is identical in both the incremental and the complete RM generation experiments, since even though the alignment probabilities are being updated, only the new data is being aligned, while the older data is left untouched. As a result, we obtain the same phrase pairs from the new data for both RM generation methods. Given that, our algorithm produces the exact same reordering model as its generation from the entire data (up to numerical accuracy).

Algorithm 1 Merging reordering tables with counts

```
1: procedure MERGE_R_TABLES(R_A,R_B)
2:     Read first lines of R_A and R_B, R_A^(1), R_B^(1)
3:     i := 1; j := 1
4:     while R_A^(i) ≠ null and R_B^(j) ≠ null do
5:         if R_A^(i) < R_B^(j) then // Compare bi-phrases
6:             R_A^(i) → R_AB
7:             i := i + 1
8:         else if R_A^(i) > R_B^(j) then
9:             R_B^(j) → R_AB
10:            j := j + 1
11:        else // Identical bi-phrases
12:            MERGE_LINES(R_A^(i), R_B^(j)) → R_AB
13:            i := i + 1; j := j + 1
14:        end if
15:    end while

    // Write the rest of the tables:
    //    at least one of them is EOF
16:    while R_A^(i) ≠ null do
17:        R_A^(i) → R_AB
18:        i := i + 1
19:    end while
20:    while R_B^(j) ≠ null do
21:        R_B^(j) → R_AB
22:        j := j + 1
23:    end while
24: end procedure
```

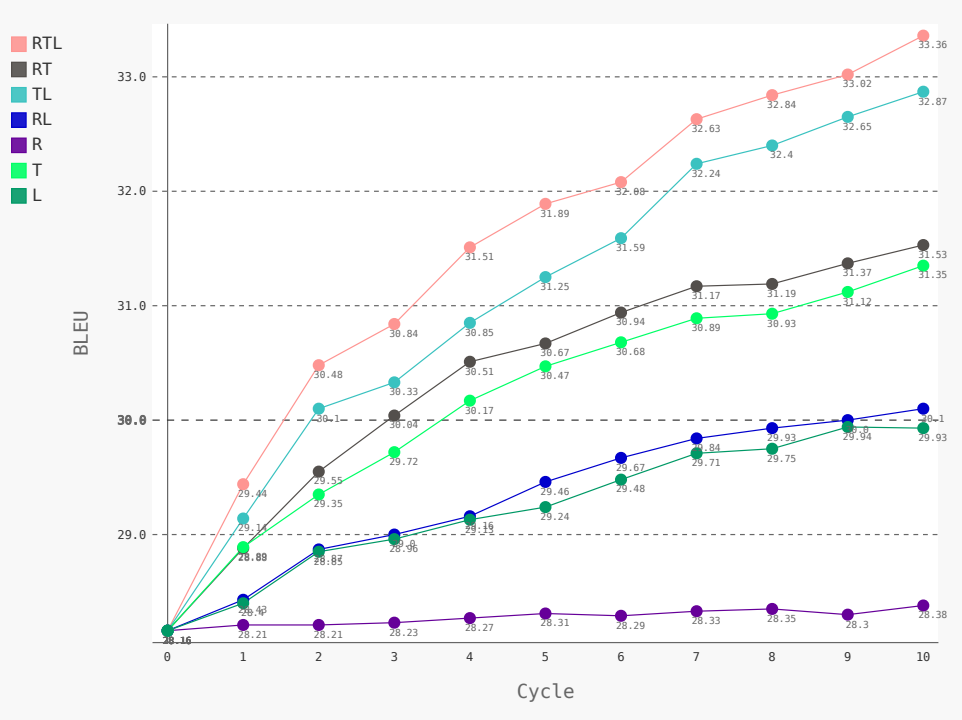


Figure 15: Translation performance (BLEU) when incrementally updating the model with additional data, over 10 update cycles, with different combinations of **R**eordering, **T**ranslation and **L**anguage models.

Translation performance

First, we demonstrate that updating the reordering table help achieving better translation quality. To that end, we compare all possible combinations of updating the three phrase-based SMT sub-models (reordering, translation and language models, denoted R, T and L, respectively). Figure 15, that includes a detailed view of Figure 10, shows the results of the experiments with each one of these combinations. From the figure we learn that: (i) the reordering model is the least important one of the three. This is consistent with prior work, e.g. [14]; (ii) updating the reordering model without updating the translation model has practically no impact on results, since new phrase pairs from the new data that are not added to the phrase table cannot be used in the translation. This is reflected in the almost flat line of experiment R, and in the very similar results of RL in comparison to L. The slight improvement in this case may be attributed to more statistics that have been accumulated for the phrase pairs that already existed in the initial data; (iii) when the translation model is updated, adding the reordering model does help, as seen in RTL vs. TL and RT vs. T.

Figure 16 shows the same experiment using transLectures data. The updates occurs over 10 cycles, each with 500 bi-sentences from the transLectures training set, and with the transLectures English-French test set of 1,363 sentences. No other parameters were changed relative to the experiment described above. As seen in the figure, updating the reordering model alongside other models is also useful in this case.

Run-time

We now compare the time necessary to train a reordering model from scratch (complete training) vs. using the suggested incremental update. For this experiment, we used the English-French Europarl corpus, with 1.96 million parallel sentences as $\mathcal{A}$ and 10,000 WIT3 sentence-pairs as $\mathcal{B}$. Other details of the WIT3 experiment setting did not change.

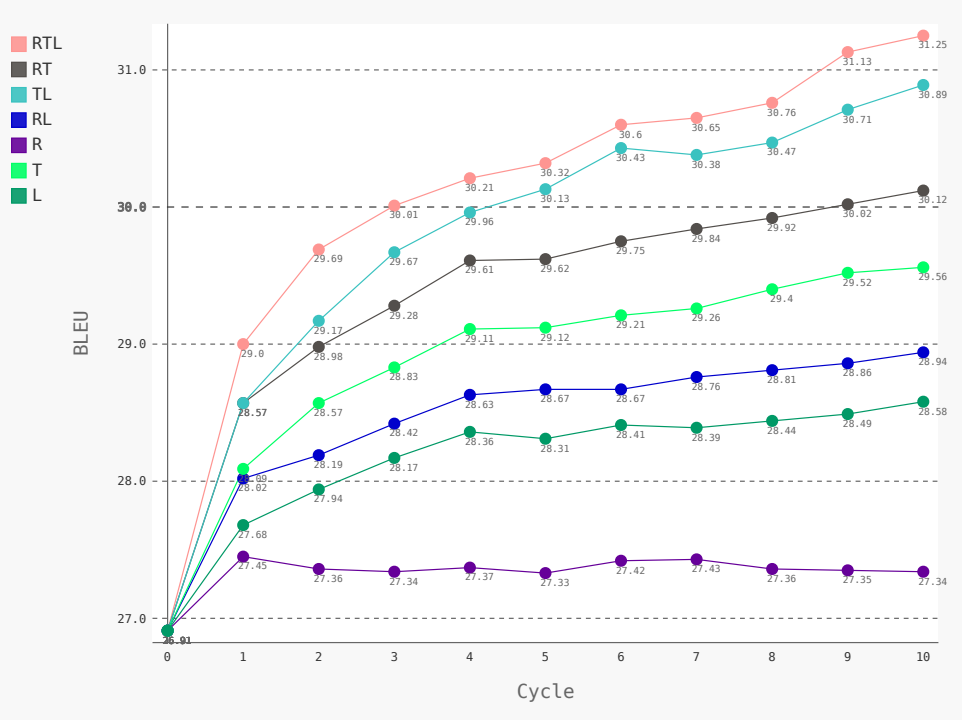


Figure 16: Translation performance (BLEU) when incrementally updating the model with additional transLectures data over 10 update cycles, with different combinations of **R**eordering, **T**ranslation and **L**anguage models.

To objectively measure the run-time of the required steps, regardless of the computer's load at the specific time of experiment, we use the Linux command *time*, summing up the *user* and *sys* times, i.e. the total CPU-time that the process spent in user or in kernel modes. All measurements were conducted on a 64-bit Centos 6.5 Linux server, with 128 GB of RAM and 2 Intel Xeon 6-core 2.50GHz CPUs.

A complete reordering model update, when using Incremental GIZA, consists of of the following two steps:

1. Extracting phrase pairs and creating a reordering file from all the data ($\mathcal{A} \cup \mathcal{B}$)
2. Creating a reordering table from the single reordering file of $\mathcal{A} \cup \mathcal{B}$

In comparison, the incremental update requires the following steps:

1. Extracting phrase pairs and creating a reordering file from the new data ($\mathcal{B}$)
2. Creating a reordering table from the reordering file of $\mathcal{B}$
3. Merging the RTs of $\mathcal{A}$ and $\mathcal{B}$

The time required for generating the complete model in our experiment was 83.6 minutes, in comparison to 17.6 minutes for the incremental one, i.e. 4.75 times faster.

We note that $\mathcal{A}$ represents a corpus of medium size, and often the initial corpus would be much larger.[8] Concerning $\mathcal{B}$, say we plan to perform daily system updates, then a set of 10,000 sentences pairs constitutes a substantial amount of data in terms of what we can expect to obtain in a single day. Hence, the time gain in actual settings may be even larger.

[8]For comparison, the rather popular MultiUN corpus [4] consists of 13.2M parallel sentence for this language pair (http://opus.lingfil.uu.se/MultiUN.php, accessed on 7 August 2014).

5.3.5 Reordering model update: conclusions

In this part we showed that updating the reordering model is useful for obtaining improved translation, even for a language pair such as English-French, where phrase movements are not very prominent (in comparison to English-Japanese, for example). We proposed a method for incrementally training this model within the Moses SMT system, which can be done much faster than a complete retrain. It thus supports more frequent SMT model updates to enable quickly benefiting from newly obtained data and user feedback and reflecting it in the system's translation.

5.4 Ordering the incremental training data

5.4.1 Introduction

Sections 5.1 and 5.3 described methods to speed up model updates given new bilingual data. The assumption was that the data is provided to us, and our goal was to quickly integrate it into the model. We now turn do address somewhat different scenarios. One scenario we may be facing is when a large amount of new training data has become available, but we cannot, for computational reasons, use it all immediately. One option is to apply *data selection* over the data, i.e. choosing which bi-sentences to use for the update. This is a common technique that is applied particularly in the *domain adaptation* line of research, where *in-domain* data is chosen from a larger pool of bi-sentences (see D3.1.3 concerning data selection). In contrast to data-selection, here we assume that all the data is relevant and are not primarily investigating reducing the size of the data, but rather scheduling its utilization by defining the order in which it will be added to the model. Another motivation is prioritizing the sentences that are sent for translation (or post-edition), or – assuming the translations are not received all at once – deciding when to feed them into the system. Hence, determining the order of adding training data can be done post-hoc, after we obtain the translations, or in advance – for sending the sentences for post-edition according to the desired order.

Below are three considerations whose impact we may wish to assess when determining the order of the data are listed below.

- **Informativeness**: We would like the data we add to contribute the most to the model. For instance, adding a seen sentence-pair may be less useful, that in contrast to a sentence with many unseen words.
- **Difficulty**: Under this consideration we would prefer adding "easier" data points first, to avoid inserting errors into the model that will be propagated to subsequent models. As we assume we are using correct (post-edited) translations, difficulty refers to alignment and not to translation. As we have shown in Section 5.2, alignment has direct impact on translation quality. For instance, presumably a sentence with many unseen words may be more difficult to align to its translation and could introduce alignment errors to the model. See Figure 17 for an example.
- **Dependency**: Here our assumption is training examples are not independent and some dependency between them exists, which we may want to consider. For instance, it may be useful to group together multiple instances of the same OOV to allow proper alignment of these words within a single mini-batch. Note that this option is probably not the one that maximizes the informativeness, as it would prefer repetitions of OOVs over new ones. Note also that this is a consideration of incremental training and would not be an issue when batch training is applied.

Hence, the above considerations may be contradicting each other. Finding the impact of each of these considerations and the preferred way to take them into account is the topic XRCE investigate in this work.

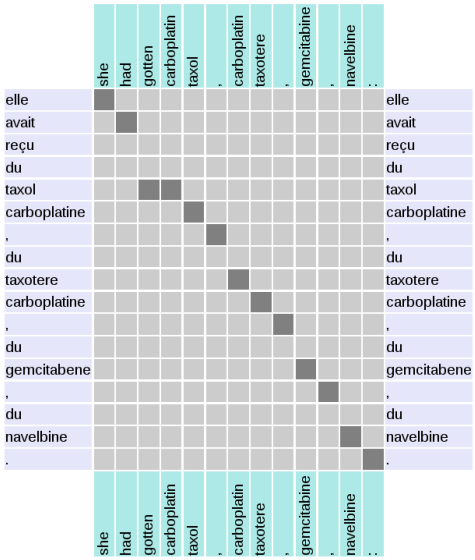


Figure 17: Word alignment of the sentences s: *"she had gotten carboplatin taxol , carboplatin taxotere , gemcitabine , navelbine :"* and t: *"elle avait reçu du taxol carboplatine , du taxotere carboplatine , du gemcitabene , du navelbine ."* Alignment errors of the unknown words *carboplatin* and *taxol* result in the erroneous translation of these words later on: *"she had gotten taxol carboplatin"*→*"elle était carboplatine carboplatine"*. The alignment is visualized here with the MT-EQuAl Toolkit [7].

5.4.2 Experiments

In the set of experiments described below, we assess the impact of two simple characteristics of the **source** sentence on translation quality. Our assumption is that the two first considerations mentioned in Section 5.4.1, namely informativeness and difficulty, correlate with these characteristics. Difficulty is assumed to correlate with: (i) the length of the sentence (ii) the number of OOV words it contains. These are two features that are known to be useful for MT confidence estimation. Informativeness is estimated by the number of OOVs in the sentence.[9] With respect to length, we assume that the longer the sentence, the more OOVs it contains. As shown in Figure 18, this assumption is generally correct. Hence, length corresponds to informativeness as well, both because long sentences contain more unseen words, and also because they augment the statistics of seen ones.

The experiments are conducted within the same setting described in Section 5.2, with the exception that the reordering model is not updated. Figure 19 shows the translation results of 10 cycles of incremental training when training sets are ordered according to different criteria:

- *min-oov*: the sentences with the minimal number of OOVs are placed first.
- *max-oov*: the sentences with the maximal number of OOVs are placed first.
- *shortest*: the sentences are ordered by their length, shorted first.
- *longest*: the sentences are ordered by their length, longest first.

The figure also shows two baselines:

- *baseline*: the training data is left as-is, without re-ordering.
- *random*: this is the average of two runs in which the data was randomly ordered.

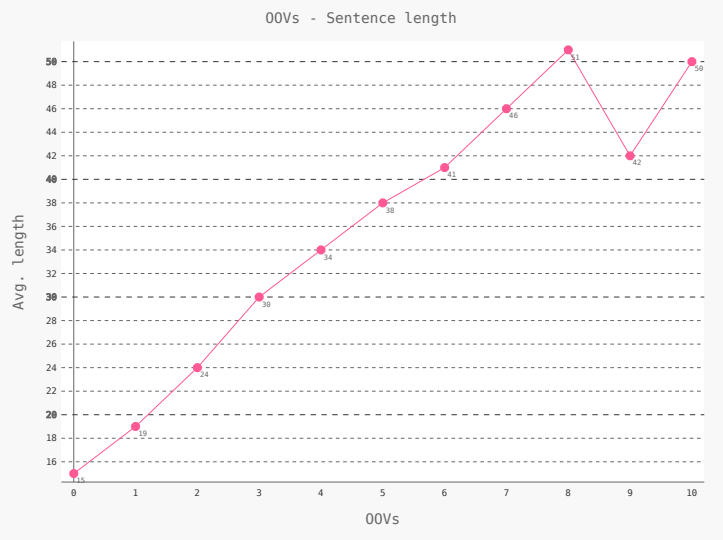


Figure 18: Sentence length as a function of the number of OOVs it contains. The average figures are rounded to the closest integer, and are shown if at least 5 sentences contained the same number of OOVs (e.g. only 2 sentences contained 12 OOVs, and we therefore do not show an average length for the value 12.)

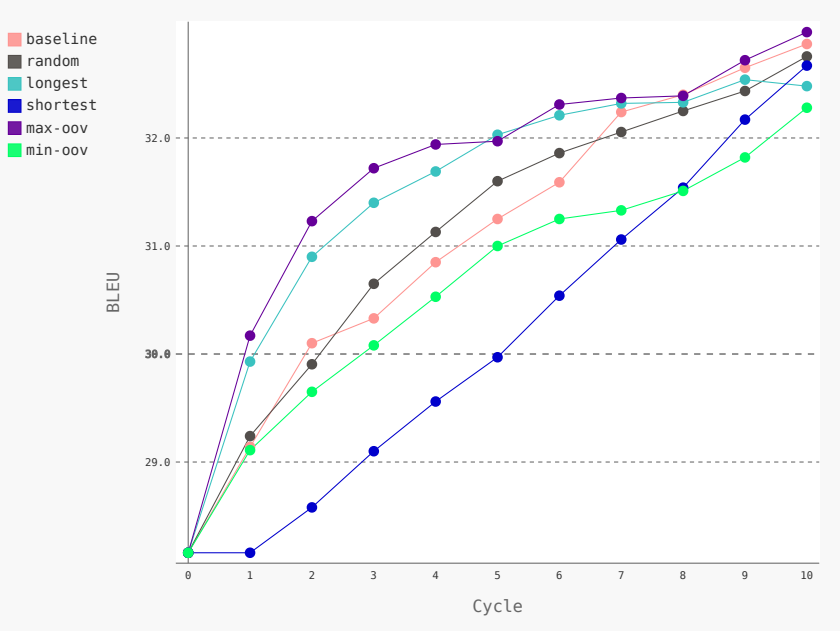


Figure 19: Translation performance of the different ordering criteria, with incremental training over 10 update cycles.

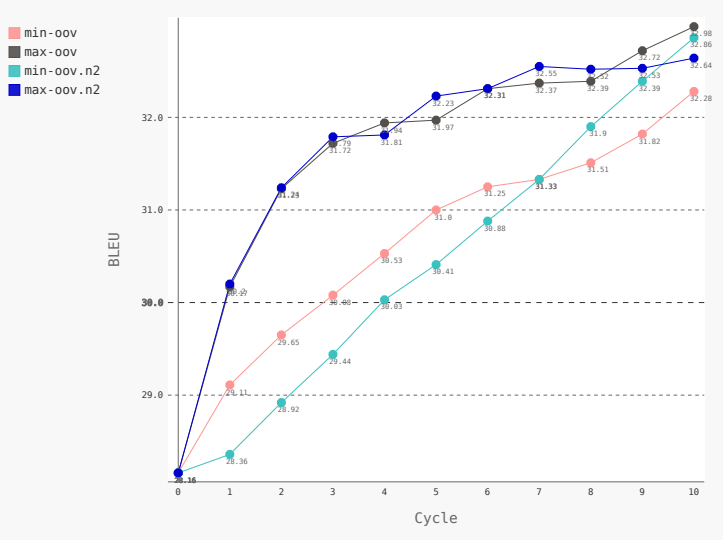


Figure 20: The impact of considering longer n-grams ($n = 2$) when ordering the data.

Considering both informativeness criteria – the length and the number of OOVs – we learn that, at least in the setting of this experiment with a small initial model, the informativeness factor plays a much more important role than difficulty, as the model gains faster improvements when we feed it with unseen words, and eventually even ends up with a better performance.[10] The significant impact of adding translations of OOVs is consistent with results shown in Section 4.6.

When the sentences are ordered from the shortest to the longest we see that in the first cycle the performance is unchanged, suggesting that these sentences contribute nothing to the model. That, in contrast to the *min-oov* order that helped even in the first cycle, even though none of the sentences in that cycle consisted of any unseen words.[11]

Next, we assess the impact of considering longer unseen n-grams when ordering, now counting also unseen bigrams and not only unigrams to determine the order of the data to add first. Figure 20 shows the results of this experiment in comparison to the *min-oov* and *max-oov* experiments shown above where $n = 1$. We see that when we order with the maximal number of OOVs first, we get pretty much the same results for $n = 2$ and $n = 1$. However, when we order with the minimal number of OOVs first, performance is even slower to catch up when $n = 2$, possibly because the sentences that are places first are even more "familiar" to the model (which has seen their bigrams), and therefore contribute less.

In the experiments described above, the number of OOVs in each sentence was counted without considering repetitions of OOVs across sentences. That is, if for example, two sentences contain the same OOV word, this word would be counted once for each one of them. We now assess updating the counts of the OOVs in the sentences online, during the ordering process: Once a sentence was selected, we remove its unseen tokens from the list of OOVs, thus decreasing the OOV counts of (some of) the remaining sentences.

The results of this experiment are shown in Figure 21. Following is a possible explanation of the results. When the sentences are ordered with the maximal number of OOVs first, the

[9] With the exception of the experiment described in Figure 20, 'OOV' refers to an unseen *unigram*.

[10] Recall that incremental GIZA uses an interpolation of statistics from the older and newer data, and does not compute the model on the entire data. Therefore, the final results of the different update options may differ.

[11] When we encounter ties in the number of OOVs or the number of tokens, we maintain the original order. Sentences without OOVs spanned over more than 5 cycles, hence, the first cycle of *min-oov* do not necessarily contains the shortest sentences.

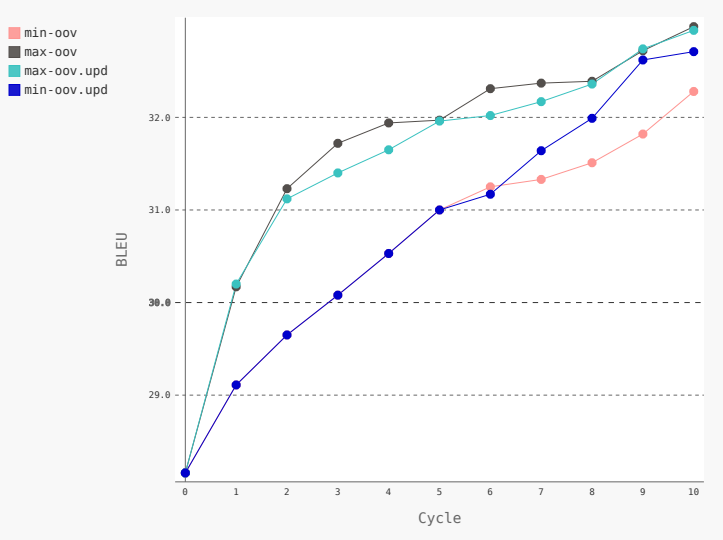


Figure 21: Ordering by the number of OOVs, when the counts are updated (*upd*). The 5 first cycles of *oov-min* and *oov-min.upd* are identical since they all contain only sentences without any OOVs.

update pushes sentences with OOVs that were seen in other sentences towards the end, as their counts are reduced. A possible result is that unseen words (with respect to the initial model) occur fewer times together, in comparison to the order without online updates. When the order is from few to many OOVs, the update causes sentences with words that were unseen by the initial model but seen by previous sentences to be pushed to the beginning, since their counts are decreased. This creates the counter-effect – putting more unseen words together. The fact that in the former case, the model performance is inferior and in the latter it is superior in comparison to the non-updated versions, may support to the dependency criterion that we suggested above.

5.4.3 Incremental training data order: conclusions

In this part of the work we propose the idea that the order of the data we use for multi-cycle incremental training affects translation performance. We have proposed several criteria to determine the order and estimated their effect through a set of experiments. The results validate our proposal and suggest that determining the order of the data to feed into the system can have significant results on intermediate models. Informativeness, manifested though both the length of a sentence and the number of OOVs it contains, stood out to be a prominent criterion, yielding faster model improvements.

6 Summary for the Duration of the Project

6.1 Fast Constrained Search

Regarding constrained search for transcription, during the first year of the project two different systems were developed. In the case of UPVLC system, user constraints were defined to deal with user supervisions at word-level whereas the RWTH system introduced a prefix-constrained search based on the open source speech recognition system RASR [18]. Along the second year, UPV implementation was extended to deal with constraints at phrase-level (sequences of consecutive words). In the same way, RWTH implementation was extended to multiple prefix-constrained search allowing prefixes not only at the beginning of a sentence but also inside a sentence. Finally, in the last period up to M36 the RWTH implementation was improved to allow multi-pass recognition. UPV algorithms have been added as features to the open source TLK ASR toolkit.

Regarding constrained search for translation, during the first year, constrained search was implemented based on two kinds of user input: an unordered set of words (bag-of-words) and an ordered sequence of words. In the second year a more involved type of interaction was considered where the user specifies pairs of source and target sequences that are correct translations of each other. All these algorithms have been implemented into the RWTH's open source translation toolkit *Jane*.

6.2 Intelligent Interaction for Transcription

During the first year of the project, the work was focused in the development and evaluation of the II approach described in Fig. 1. The first-year evaluation of this approach proved that it can be very effective for applications in which user effort is very limited as in the case of transLectures. In particular, the II approach outperformed BI for different levels of user effort mainly due to the good performance of confidence measures. However, this evaluation was carried out following an experimental setup in which the ASR models were built from scratch. In order to better fit the proposed approach into the transLectures context, experiments starting from M12 were performed using the best ASR systems developed in WP3.

During the second year, different actions were carried out in order to improve the II approach. On the one hand, confidence estimation was improved by applying a word-based naïve Bayes classifier. A deep analysis of the behaviour of confidence measures using this classifier concluded that they are very effective when the supervision effort is below 20%. In this case, relative reductions in WER between $40\% - 60\%$ can be expected whereas minor improvements were produced applying higher supervision efforts. On the other hand, different levels of interaction units were tested: word-level, phrase-level (sequences of consecutive words) and sentence-level. This evaluation showed that the user effort is better exploited when the interaction is carried out at word-level. The reason behind this behaviour is related to the better performance of confidence measures at word-level.

During the last year of the project, the developed work have mainly consisted on the evaluation of II techniques using enhanced Spanish ASR models. Also, this evaluation has been extended to the VideoLectures.NET case study using the UPVLC English ASR system developed in WP3.

6.3 Intelligent Interaction for Translation

During the first year of the project, the work was focused in the development and evaluation of the intelligent interaction described previously in Fig. 4. In this first evaluation, the sentence

was adopted as interaction unit since it was considered as the more natural way of interaction for translation. However, experiments showed that sentences may not be optimal in terms of system performance. In order to overcome this issue, interaction at word-level was explored during the second year as a first attempt of using smaller interaction units. Moreover, within this second-year period, the experimental setup was slightly modified to better fit the transLectures context. With this purpose, experiments from M12 were carried out using the best SMT systems developed in WP3 and the different approaches were tested over the scientific tests of transLectures. The experiments showed that user interaction at word-level is not as effective as in the case of sentences and other interaction units should be explored. During the last year of the project, a phrase-based approach for II was developed. The results showed that significant improvements regarding the word-level interaction were produced but not as high as supervising full sentences.

In the final period of the project, experiments on combining constrained search for translation and intelligent interaction were carried out. It had been identified before, that out-of-vocabulary words are the best candidate for user supervision, exploiting the fact that they often appear multiple times in a single video lecture. In combination with constrained search, the results showed improvements of 0.9% BLEU on English→German and 0.6% BLEU on English→Slovenian with a simulated user effort of no more than 0.4% of the text.

6.4 Incremental Training for Translation

Within Task 4.4, XRCE investigated methods to incrementally update translation model with new parallel sentences – the result of post-editions of automatic translations. In particular, we focused on finding practical ways to perform the update within realistic scenarios for transLectures. The approach we proposed is to consider two different update cycles. The first is a short-cycle, e.g. daily, in which only fast model generation tasks are performed. The second is a long-cycle in which slower tasks, including complete batch update, can be performed. In the second year of the project we developed a method, referred to as *quick updates*, to perform at the end of every short cycle. Within quick-updates, the training data is distributed between separate SMT components, enabling to perform the model-update faster by performing computationally-demanding tasks only on smaller parts of the entire data. This approach was shown to be feasible in terms of runtime and to achieve competitive translation performance in comparison to full re-training.

The quick update approach enabled incrementally updating the translation model and the language model, but the reordering model, which constitutes the third main component of typical phrase-based machine translation systems, could not be addressed in the same way within the current Moses translation system. We therefore proposed in the third year a method to incrementally update this model without building it all over again and showed that this method enables performing the reordering model update much faster (4.75 times faster in our experiment). Together with the earlier mentioned approach, this provides a way to quickly update the entire SMT model.

The multi-cycle update process is automated through a tool we completed in the third year. This tool also makes use of the previously mentioned update of the reordering table. Lastly, in the third year we also assessed the effect of different orderings of the training data on the translation performance when performing multi-cycle incremental training, and showed that the order in which the data is fed into the system can have a significant impact on the results.

References

[1] 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.

[2] Mauro Cettolo, Christian Girardi, and Marcello Federico. WIT^3: Web inventory of transcribed and translated talks. In *Proceedings of EAMT*, 2012.

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

[4] Andreas Eisele and Yu Chen. Multiun: A multilingual corpus from united nation documents. In *Proceedings of LREC*, 2010.

[5] Qin Gao, Nguyen Bach, and Stephan Vogel. A semi-supervised word alignment algorithm with partial manual alignments. In *Proceedings of The Joint Fifth Workshop on Statistical Machine Translation and MetricsMATR*, WMT '10, pages 1–10, Stroudsburg, PA, USA, 2010. Association for Computational Linguistics.

[6] Qin Gao and Stephan Vogel. Parallel implementations of word alignment tool. In *Software Engineering, Testing, and Quality Assurance for Natural Language Processing*, pages 49–57. Association for Computational Linguistics, 2008.

[7] Christian Girardi, Luisa Bentivogli, Mohammad Amin Farajian, and Marcello Federico. Mt-equal: a toolkit for human assessment of machine translation output. *COLING 2014*, page 120, 2014.

[8] Philipp Koehn. Europarl: A parallel corpus for statistical machine translation. In *Proceedings of MT Summit*, 2005.

[9] Philipp Koehn. *Statistical machine translation*. Cambridge University Press, 2009.

[10] Philipp Koehn, Amittai Axelrod, Alexandra Birch, Chris Callison-Burch, Miles Osborne, David Talbot, and Michael White. Edinburgh system description for the 2005 IWSLT speech translation evaluation. In *Proceedings of IWSLT*, pages 68–75, 2005.

[11] 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 *Proceedings of the 45th Annual Meeting of the ACL on Interactive Poster and Demonstration Sessions (ACL 2007)*, pages 177–180, 2007.

[12] Abby Levenberg, Chris Callison-Burch, and Miles Osborne. Stream-based translation models for statistical machine translation. In *Proceedings 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.

[13] Shachar Mirkin. Incrementally updating the smt reordering model. In *Proceedings of The 28th Pacific Asia Conference on Language, Information and Computing (PACLIC)*, Phuket,Thailand, 2014.

[14] Shachar Mirkin and Nicola Cancedda. Assessing quick update methods of statistical translation models. In *Proceedings of the 10th International Workshop on Spoken Language Translation (IWSLT 2013)*, pages 264–271, Heidelberg, Germany, Dec 2013.

[15] Franz Josef Och. Minimum error rate training in statistical machine translation. In *Proceedings of the 41st Annual Meeting on Association for Computational Linguistics - Volume 1 (ACL 2003)*, ACL '03, pages 160–167, 2003.

[16] Franz Josef Och and Hermann Ney. A systematic comparison of various statistical alignment models. *Computational linguistics*, 29(1):19–51, 2003.

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

[18] D. Rybach, C. Gollan, G. Heigold, B. Hoffmeister, J. Lööf, R. Schlüter, and H. Ney. The RWTH Aachen University open source speech recognition system. In *10th Annual Conference of the International Speech Communication Association*, pages 2111–2114, 2009.

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

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

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