
Some Tradeoffs in Continual Learning for Parliamentary Neural Machine Translation Systems

Rebecca Knowles
Samuel Larkin
Michel Simard
Marc Tessier
Gabriel Bernier-Colborne
Cyril Goutte
Chi-kiu Lo ... ½ ½
National Research Council Canada

Rebecca.Knowles@nrc-cnrc.gc.ca
Samuel.Larkin@nrc-cnrc.gc.ca
Michel.Simard@nrc-cnrc.gc.ca
Marc.Tessier@nrc-cnrc.gc.ca
Gabriel.Bernier-Colborne@nrc-cnrc.gc.ca
Cyril.Goutte@nrc-cnrc.gc.ca
ChiKiu.Lo@nrc-cnrc.gc.ca

Abstract

In long-term translation projects, like Parliamentary text, there is a desire to build machine translation systems that can adapt to changes over time. We implement and examine a simple approach to continual learning for neural machine translation, exploring tradeoffs between consistency, the model's ability to learn from incoming data, and the time a client would need to wait to obtain a newly trained translation system.

1 Introduction

There are many translation use cases in which translation is ongoing, i.e., new translations are being produced by translators. We experiment with approaches to using the flow of new data produced by translators to continually update neural machine translation (NMT) systems indefinitely. The intuition behind this is a desire to keep the system(s) up-to-date and optimal for the task of translation over time. This could include new topics that are being discussed, changes to terminology, use of spelling variants, or changes in translator/translation preferences; we specifically highlight terminology in this work due to the ease of measurement.

Our planned continual learning approach is to build a BASELINE-INITIAL system (trained on all data up to a fixed point in time) and iteratively fine-tune (continue training) it on chunks of new data, which we refer to as *stages*. This simulates the real-life scenario in which new translations are produced and then the goal is to use them to improve the qual-

ity of the existing initial translation system.

We use the term “continual learning” in this work, though we note that a number of different terms have been used more or less interchangeably to refer to this concept in the machine translation (MT) and machine learning literature.¹ In the context of MT, they refer to the idea of using the flow of new data to continue to train an MT system indefinitely, producing systems that are always up-to-date: that learn new terms, phrases and formulations, new concepts, changing translations (of old terms—interestingly, this suggests that while most old knowledge should be retained, some of it should be forgotten/overwritten), etc., as they appear. Within this work we will refer to these approaches as *continual learning* (hereafter CL).

The research question we address is whether a simple approach of regularly finetuning a base model works successfully as a CL approach to NMT, especially in a Parliamentary setting, where the domain may evolve over time, but is not expected to suddenly change completely. We describe proof-

¹Those include “continuous learning”, “lifelong learning”, “translation project adaptation”, as well as the related concepts of “online learning” and “incremental learning” or “incremental updating”.

of-concept experiments, in an idealized setting, but with real data, where we compare with several baselines. While our work examines a realistic use case, it is limited by our focus on one language pair (English–French) and one domain (parliamentary text). We examine tradeoffs with respect to questions of various measures of model performance, and note areas where more study is needed to determine the usefulness of these approaches. We also provide a brief discussion of the technical infrastructure that would be required to implement these approaches in practice, with considerations around feasibility and potential costs and risks.

2 Related Work

We provide a brief overview of related CL literature from machine learning (more broadly) and machine translation (more specifically).

2.1 CL for Natural Language Processing

In their survey on CL techniques in natural language processing (NLP), [Biesialska et al. \(2020, p. 6524\)](#) define CL as “a machine learning paradigm, whose objective is to adaptively learn across time by leveraging previously learned tasks to improve generalization for future tasks.” Language usage and topics of interest change over time due to various linguistic and social processes and, as a result, machine learning models at the heart of NLP applications tend to become less accurate or stale. Periodically training new models, using data that better reflects the changing distribution of data, is an effective but often highly inefficient and costly solution. This motivates the need to find ways to “continue” the training of NLP models as new data becomes available. In practice, however, existing models often struggle to adapt to new information while simultaneously retaining previously learned knowledge, a problem which can eventually lead to *catastrophic forgetting*, where the improvement on a new task or new data set simultaneously results in a dramatic degradation in quality on the original training task or data ([Goodfellow et al., 2014](#)). This gives rise to the *stability-plasticity dilemma* discussed in [Biesialska et al. \(2020\)](#): the main challenge in CL is to strike a balance between the model’s *stability* (its ability to retain prior knowledge) and its *plasticity* (its ability to adapt to new knowledge). That survey highlights three major approaches to this chal-

lenge: *rehearsal* approaches, where older training samples are kept for each task and periodically revisited while updating a model; *regularization* approaches, which modify the learning objective to aid knowledge consolidation while learning subsequent tasks, for example by slowing down the learning of parameters deemed important for previous tasks; and *architectural* approaches, where changes are made to a model’s architecture, making it possible to introduce task-specific parameters and isolate or better control their effects.

2.2 CL for Machine Translation

In machine translation and computer-aided translation, updating MT models based on new translation data has been a recurring topic. This differs from the broader definition of CL, where the task itself may change: here the task of translation remains the same but the data distribution changes in potentially unpredictable ways. One type of data that is of particular interest is professional translator feedback in the form of post-edited MT output. [Cettolo et al. \(2014\)](#)—in the phrase-based statistical MT paradigm—proposed an approach that they call “translation project adaptation.” In their setting, a translator performed post-editing, and this post-edited data was then used to adapt the MT system for the future, iteratively improving the accuracy of translations. This concept was later adapted to the neural machine translation (NMT) setting by numerous researchers. For example, [Álvaro Peris and Casacuberta \(2019\)](#) perform updates by finetuning the parameters of a NMT model with every new post-edited sentence (in simulation), yielding better quality translations than the base model and reducing the human effort required to correct the system’s output. [Kothur et al. \(2018\)](#) and [Knowles \(2019\)](#) find similar results in simulations of finetuning on individual sentences and document-specific dictionaries. As in other applications of CL, catastrophic forgetting is a major concern.

The three major approaches to CL have also been applied to MT by researchers, including rehearsal approaches ([Chu et al., 2017](#); [Zhang et al., 2019](#); [Bengio et al., 2009, i.a.](#)), regularization approaches ([Khayrallah et al., 2018](#); [Kirkpatrick et al., 2017](#); [Cao et al., 2021, i.a.](#)), and architectural approaches ([Freitag and Al-Onaizan, 2016](#); [Gu and Feng, 2020](#); [Gu et al., 2022](#); [Li et al., 2020](#); [Bapna](#)

and Firat, 2019; Wang et al., 2022, i.a.).

In this paper, we do not explore any of these specifically proposed solutions to the catastrophic forgetting problem; instead we simply aim to be alert to the risks (by measuring performance on new data and held-out original data). Since we examine a scenario where we do expect gradual change over time, there are likely some things that *should* be forgotten over time as others are learned (e.g., new preferred terminology or translations).

3 Data Setup

The general data setup for our experiments is designed to mimic in a controlled fashion a real-world scenario in which there exists a backlog of professional translations, and an incoming sequence of additional new translation pairs that are generated each day by translators. Translations are collected regularly and used to incrementally update the MT model, which is then used to produce future translations. We call each such collection period a “stage”.

All experiments are based on English–French or French–English data from the proceedings of the debates (a.k.a. “Hansard”) in the Canadian House of Commons, one of the two chambers of the Canadian Parliament. This data ranges in time from 2006-04-03 to 2023-09-29, is segmented into sentences, and is timestamped so it can be ordered sequentially, with 5-minute precision.² We explain here how we use this data for both hyperparameter-tuning experiments and full data experiments where we train and test the continual learning approach and several baselines: (1) a BASELINE-INITIAL system (trained in a standard, non-CL, manner on all data up to an initial stopping point in time); (2) a BASELINE-FINAL system (trained in a standard, non-CL, manner on all data up to a final stopping point in time); (3) a BASELINE-RECENCY model (a stronger BASELINE, specially finetuned on the CL data).

3.1 Full Experiment Data

The bottom portion of Fig. 1 shows how we have set up the data for our full experiments, while the top portion shows how the subset of data used for the hyperparameter tuning described in this work over-

laps with the full data. For building the BASELINE-INITIAL system, we use all Hansard data from 2006-04-03 until (but excluding) 2021-11-22 ($A + B$ in Fig. 1); this last date coincides with the beginning of Canada’s 44th Parliament. It is partitioned into training (A), development, and test splits (*baseline-initial-dev* and *baseline-initial-test*, sampled from B). Development and test splits for this model are 2000 sentences randomly sampled from the most recent 40000 pairs of sentences (B in Fig. 1); whatever remains is added to the training data. The *baseline-initial-test* data is also used for examining the stability of the CL systems.

All data that falls between 2021-11-22 and 2022-10-24 (inclusive; $C + D$ in Fig. 1) is divided into CL “stages”. We picked a fixed stage size of 3000 sentences, close to the average number of sentences per day in the Hansard (2904).³ Due to the small size of the stages, we choose a CL approach where we do not use a development set for early stopping; instead we use a fixed learning rate and number of epochs for all stage training. These 3000-sentence stages are used iteratively as test and then training data in CL; once a stage has been trained on, it is never again used as test.

Data for the BASELINE-FINAL system is built analogously to the BASELINE-INITIAL system, using all data from 2006-04-03 until 2022-10-24 ($A + B + C + D$ in Fig. 1), including part of the data from 2022-10-24. Again, we partition this $A + B + C + D$ data into *baseline-final-train*, *baseline-final-dev* and *baseline-final-test* by uniformly sampling 2000 sentences each for the latter two from the most recent 40000 sentences (D in Fig. 1).

The rest of the data from 2022-10-24—i.e., that which is not in set D —is included in the *epilogue* (E in Fig. 1). In practice, for the remainder of this paper, when we refer to *epilogue-test* data we only use the first 40000 lines of the *epilogue* data.

See Table 1 and Appendix A for more information on data set sizes.

3.2 Hyperparameter Tuning Data

Before experimenting on the full data set, we need to select hyperparameters. We use fixed values for

²Re. duplicates: in order to evaluate system performance in a realistic usage scenario, duplicates or sentences that appear in training are not removed from test and tuning sets (Appendix A). This allows us to monitor the “translation memory effects” in our systems.

³In this simplified, idealized setting, one day’s text may spread across multiple stages, and a stage may include several days. Future work may experiment with training on varying stage sizes.

Figure 1: Data splits for hyperparameter tuning and full experiments. In the Full experiment data split, the BASELINE-INITIAL system is trained on parts A and B ; the BASELINE-FINAL system is trained on parts A ; B ; C and D , but excluding *baseline-initial-test*. For both these BASELINE systems, tuning and test sets (2000 sentence pairs each) are sampled from parts B and D respectively; all remaining data from each set is used for training. Starting from the BASELINE-INITIAL system, CL systems are trained by iteratively fine-tuning on data “stages”; each stage is used entirely for training or testing (with no held-out tuning set). The *epilogue* (part E) is used for testing only. The Hyperparameter (HP) Tuning Split (at the top) is structured similarly.

the learning rate and number of epochs during CL, but in order to properly choose these hyperparameters without train/test set contamination, the hyperparameter tuning data must be separate from the data we use for our final CL experiments. For this, we create a second data split, with the same structure as described above, but using data entirely contained within that set’s *baseline-initial* data ($A + B$ in Fig. 1). We call this the *Hyperparameter Tuning* data set: HP data for short. In the HP data, the CL portion begins with the second session of the 43rd Parliament, on 2020-09-23, and contains only 16 stages of 3000 sentences, ending on 2020-10-22. The HP data is shown in the top portion of Fig. 1. Parts $A^0; \dots; E^0$ in that data serve analogous functions to $A; \dots; E$ in the full data.

The *HP-epilogue-test* set consists of 4000 sentences sampled from the 40000 sentences (approximately 14 days between 2020-10-22 and 2020-11-06) immediately following the HP CL data, rather than simply the sentences immediately following the end of the CL data as is done in our full experiments.

4 Performance Evaluation

We are interested in three main types of performance evaluation: plasticity (improvement on new data), stability (maintaining high performance on past data), and volatility (whether the translations

change dramatically or incrementally between models). The goal for CL is high plasticity, high stability (i.e., no catastrophic forgetting), and low volatility.

We study plasticity on two data sets: the *epilogue-test* test set (used only for testing and never in training or parameter setting) and the sequence of intermediate test set stages. On the *epilogue-test*, we measure translation quality (using automatic metrics) of the output produced by the BASELINE-INITIAL system, each incremental CL system, and the BASELINE-FINAL or BASELINE-RECENCY system. We can compare these scores directly.

We also consider a *stage-wise evaluation* representative of real-life applications. For this evaluation, a stage is initially used as a test set, and then the CL system trains on it, testing on the subsequent stage, until reaching the *epilogue*; a stage is never again used for testing after it has been trained on. While we can compare the CL system with each of the BASELINES on each stage, we cannot directly compare the scores of the *stages* to one another (they are different test sets and most automatic metrics are not directly comparable across test sets). Instead, we compare the *difference* in metric score between the BASELINE-INITIAL system and any systems of interest. We draw an idealized line between 0 at the start of CL to the difference between BASELINE-FINAL and BASELINE-INITIAL (measured on the

Data for:	Hyperparameter Tuning			Full Experiments		
	Train	Tune	Test	Train	Tune	Test
Baseline systems						
BASELINE-INITIAL	4494960	2000	2000	4880109	2000	2000
BASELINE-FINAL	4540960	2000	2000	5262109	2000	2000
Continual learning						
CL initial	<i>(same data as BASELINE-INITIAL)</i>			<i>(same data as BASELINE-INITIAL)</i>		
per stage	3000	–	3000	3000	–	3000
			<i>("next" stage)</i>			<i>("next" stage)</i>
CL total	48000	–	45000	384000	–	381000
	<i>(16 stages)</i>			<i>(128 stages)</i>		
<i>epilogue-test set</i>	–	–	4000	–	–	367653

Table 1: Number of sentence pairs in hyperparameter tuning and full data sets.

epilogue-test) at the end of CL.

To measure stability (and whether catastrophic forgetting occurs), we use the held-out *baseline-initial-test* data set. If catastrophic forgetting occurs, we might expect to see major decreases in quality on this test, whereas if the CL approach is successful, we might expect maintained quality (or small variation; there may be changes in terminology or other such domain evolution features). This measurement will also be done with automatic MT metrics.

We examine volatility during our HP search experiments using the *HP-epilogue-test* set. A highly volatile system would see major changes from system to system; a less volatile system would likely see most translations remain similar to one another.

5 Hyperparameter Tuning Experiments

Even using the simple CL approach we have selected, we need to set some hyperparameters. We limit these to the learning rate (LR) and the number of epochs. In this section we describe the setup of our hyperparameter tuning experiments. We describe BASELINE models built specifically for HP tuning in Section 5.1, the tuning procedure in Section 5.2, performance over time in Section 5.3, and volatility in Section 5.4. We note that for our HP tuning data set, we are considering a smaller range in time than in our full data set.

⁴BLEU scores (Papineni et al., 2002) are computed using sacrebleu (Post, 2018) with a signature of nrefs: 1|case: mixed|eff: no|tok: 13a|smooth: exp|version: 2.3.2.

5.1 Baseline Models

We train the HP-BASELINE-INITIAL model on the *HP-baseline-initial-train* data, with early stopping done using the *HP-baseline-initial-dev* set. The HP-BASELINE-FINAL is trained using the same settings as the HP-BASELINE-INITIAL, using the relevant HP data described in Fig. 1. We build a stronger BASELINE with a recency bias, an oracle finetuned model: HP-BASELINE-RECENCY. This model is the HP-BASELINE-INITIAL finetuned on all CL data, selected from a grid search of hyperparameters based on oracle performance (BLEU score) on the *HP-epilogue*.

All models were trained using Sockeye (Hieber et al., 2022), on 4 Tesla V100-SXM2-32GB GPUs. Appendix B describes training in more detail.

5.2 Hyperparameter Tuning

Using the HP CL data set, we experiment with 6 different LRs between $1 \cdot 10^{-6}$ and $3 \cdot 10^{-4}$, and 8 values for the number of epochs, from 1 to 2^7 . For each experiment, these are fixed to avoid the risk of using a validation set (e.g., for early-stopping) that is too small to draw accurate conclusions from. When training on a stage is complete, the resulting model is used for translation of the test sets and as the starting model parameters for finetuning using the subsequent stage’s data. For each setting, we run experiments on the sixteen stages, then compute BLEU score⁴ gain at each stage, i.e., the difference between

the scores obtained with the CL system and the HP-BASELINE-INITIAL system. For a CL system fine-tuned on stage n , it can be tested on the held-out HP-BASELINE-INITIAL test data, stage $n + 1$, and the held-out *epilogue-test* data. Whenever we compare scores of two or more systems, we compare them on the same test set.

Rather than a single stand-out pair of hyperparameters from our grid search, we found a cluster of similarly-performing ones. From this, we chose a high-performing (high median BLEU score gain over stages, relatively low standard deviation) set of HPs with a low number of epochs (to decrease training time and cost).⁵ We selected the following HPs for our full experiments: for EN-FR, LR= $1.0 \cdot 10^{-5}$ and 8 epochs, and for FR-EN, LR= $3.0 \cdot 10^{-6}$ and 8 epochs. We will perform our initial analyses using these parameters. We later also compare against other settings. See Appendix C for additional details on HP selection.

5.3 Performance Over Time

Figure 2: Empirical incremental (EN-FR) evaluation on *HP-epilogue-test* set after training on each stage.

Fig. 2 shows incremental evaluation on the *HP-epilogue-test* set. The “selected CL” HPs were selected as described above. For the “slow CL” and “catastrophic CL” we chose a fixed LR with different numbers of epochs. For the sake of readability, we do not show the *worst* catastrophic forgetting model, as it drops off very quickly and would dominate the plot. The catastrophic forgetting sce-

nario did occur, as expected, when the combination of LR and epochs was too high, presumably leading to overfitting on the stage training data. Both our “selected” and “slow” CL models outperform the HP-BASELINE-FINAL model but do not reach the score of the HP-BASELINE-RECENCY.

5.4 Volatility

We also want to know whether the translations are changing substantially after each stage of training, or whether they remain generally consistent while capturing useful changes. This is likely relevant to the translator experience; highly volatile systems, where the translations of similar texts differ greatly from day to day, may reasonably lose the trust of translators or otherwise cause frustration. We examine this using the 4000-line *HP-epilogue-test* set.

Using the “selected” CL hyperparameter settings for EN-FR, we look at pairs of outputs in sequence to check how many translations change. For example, comparing the HP-BASELINE-INITIAL model to the model trained on the first stage, 1244 lines differ in their translations, while 2756 remain identical. Of all the pairs of models we examine, this is the greatest number of differing lines; most range between 1000 and 1100. Next we compute BLEU scores between the pairs of models, on the sets of sentences whose output translations differ. We treat the earlier model’s output as the “reference” and the newer model’s output as the hypothesis. These BLEU scores range from 79.5 to 82.4, indicating very high overlap between these sentences, and a visual inspection confirms this; the CL models exhibit low volatility. Performing the same tests using the “catastrophic” settings from Fig. 2 (which were not even the worst case we observed), the number of sentences that differ between adjacent models ranges from 1812 to 2056, and the BLEU scores range from 76.3 to 79.0, indicating higher volatility. If we consider the worst “catastrophic” CL (LR of $3.0 \cdot 10^{-4}$ and 128 epochs), this is even more extreme: 3754 to 3892 lines of output differ (meaning only 246 to 108 lines remain the same) and the BLEU scores between them drop to between 30.8 and 43.7.

Thus we find that with a strong set of HPs, we observe relatively low levels of volatility, supporting our decision to focus more on other aspects in

⁵There are a number of different ways one could choose between these, this heuristic is only one possible approach.

selecting the HPs. However, it would still be useful to verify through user studies that translators using the system find this behaviour satisfactory.

6 Full Data Experiments

Using the full experimental data set allows us to examine CL performance over a longer timespan and to see whether it remains consistent,⁶ and whether we need to build in additional safeguards, e.g. the ability to roll back to an earlier model if a particular update degrades quality. We also examine the various measures of plasticity, stability, and volatility over a larger timespan.

We build BASELINE models using the same configurations as the HP-BASELINES, but using the larger data set. For the BASELINE-RECENCY model, we finetuned the BASELINE-INITIAL model on all but the last stage of the CL data, using a lower initial LR and using the last stage of CL as a validation set for early stopping rather than an oracle.

We begin with the same preliminary experiments that we produced for HP selection, which we describe in Section 6.1. We then examine questions of novel vocabulary in Section 6.2.

6.1 General Analysis

We examine the results obtained with the HPs we selected for EN-FR: LR=1:0 10^{-5} and 8 epochs; finding that our heuristics seem to have led us to select hyperparameters that are too aggressive, we examine a lower LR version as well: LR=3:0 10^{-6} and 8 epochs.⁷ As we see in Figs. 3a (original HPs) and 3b (lower LR), only the lower LR HP successfully outperforms the BASELINE-FINAL model on the *epilogue-test* data and has a clear upward trajectory; neither outperform the BASELINE-RECENCY model, though both outperform BASELINE-INITIAL. In Fig. 3c, the stage-wise evaluation, we see the relative performance bouncing up and down, sometimes dipping below the BASELINE-INITIAL. In contrast, with the lower LR, Fig. 3d shows an upward trend

and less severe drops below the BASELINE.⁸ The lower LR also shows better stability performance (see Appendix D.2). This suggests that the lower LR set of HPs is better, on the basis of our earlier goals. However, there is a tradeoff: this system is slower to learn translations of new terms (see Section 6.2).

6.2 New Terminology

As Kothur et al. (2018) suggest finetuning NMT on new revised translations as an effective way of incorporating new vocabulary, an interesting case for CL is the appearance of new words or phrases that appear in the CL data but that did not appear in the BASELINE-INITIAL system's training data ("out-of-vocabulary" or OOV). We examined a number of such terms (see examples in Table 2), both single words and multi-word terms that appeared for the first time in parliamentary proceedings between November 2021 and October 2022. Where terms have multiple forms (e.g., plurals, inflected forms), we manually clustered these together as appropriate. We then assessed to what degree the BASELINE-INITIAL and CL systems produced the correct translations, as found either in *Terminus Plus*⁹ or in the Parliament translators' "Aide-mémoire" (tip sheet), where translators consign recommendations for commonly encountered translation problems.

In Fig. 4, we show a visual representation of terms and their translations over time. This allows us to see how it often takes repeated instances of a term and its translation appearing in training stages before it starts to appear in the CL output. Additionally, this highlights the difference between the two CL models, as we see that the one with the lower LR is slower to adapt to these new terms (this is the broader trend across terms examined).

NMT systems are sometimes capable of handling OOV proper names correctly, insofar as they are written similarly in the source and target languages. However, casing differences can be a prob-

⁶We do note that there is an upper limit on performance; i.e., we cannot expect BLEU scores or BLEU score differences to continue increasing forever, as they range from 0-100. Additionally, natural variation in language means that a "perfect" BLEU score of 100 is not generally a desired or achievable goal. The same is generally true of other reference-based automatic metrics.

⁷Results in the FR-EN direction show similar trends given the same lower LR HP settings; we include these figures in Appendix D.1.

⁸We see a similar pair of trends when measuring with COMET (Rei et al., 2022) – version unbabel -comet=2.2.2 with model Unbabel/wmt22-comet-da – instead of BLEU or chrF, with our initially selected hyperparameter settings even drifting below the baseline, while the lower LR shows a positive upward trend; see Fig. 10 in Appendix D.

⁹Terminus Plus is the Government of Canada's terminology and linguistic data bank: <https://www.btb.termimplus.gc.ca> (Bernier-Colborne et al., 2017).

(a) 1.0 10^{-5} LR and 8 epochs

(b) 3.0 10^{-6} LR and 8 epochs

(c) 1.0 10^{-5} LR and 8 epochs

(d) 3.0 10^{-6} LR and 8 epochs

Figure 3: **Top:** EN-FR incremental evaluation on the *epilogue-test* set after training on each stage of data, compared against BASELINES, for two LR. **Bottom:** EN-FR stage-wise evaluation showing relative performance (BLEU between CL model and BASELINE-INITIAL model) on individual CL stages.

EN Term	FR Term	EN matches seg.	stg.	FR reference consistency	FR MT Accuracy base.-init.	CL	"CL Priming" seg.	stg.
advergames	publidingvertisements ~	9	1	100%	0%	0%	–	–
crypto(-)asset	crypto(-)actif ~	54	4	100%	0%	4%	52	2
divisive	clivant(e)(s)	193	64	11%	0%	1%	166	48
freedom convoy	convoi pour la liberte •	80	33	75%	0%	61%	10	6
greedflation	cupidiflation ~ •	43	11	72%	0%	0%	–	–
omicron	Omicron •	355	64	98%	3%	55%	50	8
vaccine passport(s)	passeport(s) vaccin(al,aux) ~	61	36	100%	64%	93%	0	0

Table 2: Examples of English terms encountered in CL data, along with the recommended translation in French. Translations marked ~ are those prescribed by Termium Plus; those marked • are those prescribed in the Parliament translators' "Aide-mémoire". *Reference consistency* is the percentage of English term occurrences for which the corresponding French term appears in the reference translation. *MT Accuracy* is computed over the full CL data, as the percentage of the time that the MT system (either BASELINE-INITIAL or the CL system with hyperparameters 1.0 10^{-5} LR and 8 epochs) produces correct output for a given source term. "CL Priming" refers to the amount of exposure of the CL system to a new term before it produces a first correct translation for that term; it is reported in number of segments and number of stages (when the system never successfully translates a term, this is indicated by "–").

Figure 4: Terminology learning over time. The area of each outer grey circle represents the number of times the source term appeared in a stage; the darker overlapping inner circle represents the number of times its translation appeared in the reference, BASELINE system, and two CL systems (tested stage-wise), respectively. Each horizontal axis ranges from the first occurrence of the given term in the CL data to the last.

lem for the BASELINE-INITIAL system, as in the COVID variant *omicron*, which must be written with an initial capital in French. The CL system eventually catches this difference after 8 stages, containing 50 occurrences. In some cases, if multiple occurrences of a new term appear in the first few stages, CL can respond more quickly. For example, it appears to learn the translation of *cryptoasset* after just two stages, containing 51 occurrences of the term. When all occurrences of a new term occur within the same stage (e.g., *advergames*), the CL system may learn the proposed translation, but never gets the chance to demonstrate this. CL treatment of a new term can be affected if its translation is not consistent across all stages. For example, even though the term *greedflation* appears 43 times in 11 stages, CL fails to capture its proposed translation (*cupidiflation*), possibly because this is not used systematically in the reference (72%).¹⁰

For multi-word terms, the effectiveness of the BASELINE-INITIAL and CL systems depends in part on the term's degree of lexicalization (or fixedness, or more generally termhood). For example, for *vaccine passport* or *vaccine passports*, the BASELINE-

INITIAL system does produce the correct translation (*passeport vaccinal* or *passeports vaccinaux*) 64% of the time. But it also produces various alternatives, such as *passeport de vaccins*, *passeports pour les vaccins*, etc. In contrast, the CL system gets it right 93% of the time. In another instance, the BASELINE-INITIAL system systematically fails to translate *freedom convoy* as *convoi pour la liberté*, preferring *convoi de la liberté*. CL picks up the correct form after six stages, containing 10 occurrences, and from there produces the prescribed translation for 70% of the remaining occurrences (61% global accuracy). Other terms with a relatively complex translation, for example *rent-to-own programs* – *programmes de location avec option d'achat* and *housing accelerator fund* – *fonds pour accélérer la construction de logements* are particularly difficult for the baseline-initial system; in this case, CL eventually produces the correct translation, sometimes quite rapidly (after just two stages for *rent-to-own programs*), sometimes more slowly (after eight stages for *housing accelerator fund*).

¹⁰Regarding the consistency of reference translations, it should be noted that in these experiments, systems were trained and tested without regard for the original language in which texts were initially produced. Therefore, some apparently inconsistent translations may instead be reflective of usage variations in the source language.

7 Implementation and Future Work

All of these experiments were performed in simulation, and there are a number of additional factors that would need to be taken into account in order to implement and run CL in a practical setting. These include data collection (live rather than after full publication), monitoring (automatically or manually monitoring performance over time), roll-back (to earlier versions before restarting CL), and integration into computer-aided translation tools.

All of these considerations come with costs and risks. There are financial costs in terms of hardware (e.g., GPU purchases or cloud costs) as well as the maintenance and monitoring of the system. There are risks in terms of volatility, instability, and the possibility for catastrophic forgetting. Though our setup has aimed to mitigate some of those risks, a live system would need to incorporate monitoring and failsafes for them. Finally, these costs and risks would need to be weighed against the level of improvements observed by translators in using the tool: that is, are the benefits significant enough to warrant that investment in hardware, system development, and maintenance, as compared to other less-dynamic approaches. This requires validation through user studies, which will enable us to better understand whether the desired goals are being achieved, such as improving translations of novel terms or learning from translator corrections, in addition to maintaining high translation quality. In particular, it will be important to examine whether the changes we observe using automatic metrics are actually significant to users of the tool, and whether they find them to be beneficial or not. We will be interested to explore the impacts of plasticity, stability, and volatility on translator satisfaction.

As we noted, our initial hyperparameter selection heuristics did not result in ideal performance. Future work could more closely examine how to select the hyperparameters. Ideally this would be done in consultation with the intended users of the system, to ensure that we focus on the preferred properties of the system (whether that be consistency, rapid adaptation to new terminology, or other factors). For terminology, it may also be worth comparing against dictionary-based methods, though (Knowles et al., 2023) also note some challenges to that approach, focusing on this same domain.

8 Conclusion

In conclusion, we have demonstrated in simulation that a simple approach to CL can be effectively applied to Parliamentary machine translation systems. We find that choosing a good set of hyperparameters enables us to build models with high levels of plasticity and stability, with low levels of volatility. However, we also demonstrate tradeoffs: time, plasticity, and performance. Our approach provides incremental improvements over time, but a client who is willing to wait longer for a better system may prefer to opt for finetuning on a larger amount of recent data rather than these incremental improvements. Similarly, we see a tradeoff between rapidly learning new vocabulary and the overall performance. While we have measured the success of our models using automatic metrics, future work will also be needed to have human translators provide evaluation and feedback.

Acknowledgments

We wish to thank the parliamentary translation team and all our partners at the Canadian government's Translation Bureau, without whom this work would not have been possible.

References

- Bapna, A. and Firat, O. (2019). Non-parametric adaptation for neural machine translation. In Burstein, J., Doran, C., and Solorio, T., editors, *Proceedings of the 2019 Conference of the North American Chapter of the Association for Computational Linguistics: Human Language Technologies, Volume 1 (Long and Short Papers)*, pages 1921–1931, Minneapolis, Minnesota. Association for Computational Linguistics.
- Bengio, Y., Louradour, J., Collobert, R., and Weston, J. (2009). Curriculum learning. In *Proceedings of the 26th annual international conference on machine learning*, pages 41–48.
- Bernier-Colborne, G., Barrière, C., and Ménard, P. A. (2017). Fine-grained domain classification of text using TERMIUM plus. In Frontini, F., Grčić Simeunović, L., Vintar, Š., Khan, A. F., and Parvisi, A., editors, *Proceedings of Language, Ontology, Terminology and Knowledge Structures Workshop (LOTKS 2017)*, Montpellier, France. Association for Computational Linguistics.

- Biesialska, M., Biesialska, K., and Costa-jussà, M. R. (2020). Continual lifelong learning in natural language processing: A survey. In Scott, D., Bel, N., and Zong, C., editors, *Proceedings of the 28th International Conference on Computational Linguistics*, pages 6523–6541, Barcelona, Spain (Online). International Committee on Computational Linguistics.
- Cao, Y., Wei, H.-R., Chen, B., and Wan, X. (2021). Continual learning for neural machine translation. In Toutanova, K., Rumshisky, A., Zettlemoyer, L., Hakkani-Tur, D., Beltagy, I., Bethard, S., Cotterell, R., Chakraborty, T., and Zhou, Y., editors, *Proceedings of the 2021 Conference of the North American Chapter of the Association for Computational Linguistics: Human Language Technologies*, pages 3964–3974, Online. Association for Computational Linguistics.
- Cettolo, M., Bertoldi, N., Federico, M., Schwenk, H., Barraud, L., and Servan, C. (2014). Translation project adaptation for mt-enhanced computer assisted translation. *Machine Translation*, 28(2):127–150.
- Chu, C., Dabre, R., and Kurohashi, S. (2017). An empirical comparison of domain adaptation methods for neural machine translation. In Barzilay, R. and Kan, M.-Y., editors, *Proceedings of the 55th Annual Meeting of the Association for Computational Linguistics (Volume 2: Short Papers)*, pages 385–391, Vancouver, Canada. Association for Computational Linguistics.
- Freitag, M. and Al-Onaizan, Y. (2016). Fast domain adaptation for neural machine translation. *ArXiv*, abs/1612.06897.
- Goodfellow, I. J., Mirza, M., Xiao, D., Courville, A. C., and Bengio, Y. (2014). An empirical investigation of catastrophic forgetting in gradient-based neural networks. In Bengio, Y. and LeCun, Y., editors, *2nd International Conference on Learning Representations, ICLR 2014, Banff, AB, Canada, April 14-16, 2014, Conference Track Proceedings*.
- Gu, S. and Feng, Y. (2020). Investigating catastrophic forgetting during continual training for neural machine translation. In Scott, D., Bel, N., and Zong, C., editors, *Proceedings of the 28th International Conference on Computational Linguistics*, pages 4315–4326, Barcelona, Spain (Online). International Committee on Computational Linguistics.
- Gu, S., Hu, B., and Feng, Y. (2022). Continual learning of neural machine translation within low forgetting risk regions. In Goldberg, Y., Kozareva, Z., and Zhang, Y., editors, *Proceedings of the 2022 Conference on Empirical Methods in Natural Language Processing*, pages 1707–1718, Abu Dhabi, United Arab Emirates. Association for Computational Linguistics.
- Hieber, F., Denkowski, M., Domhan, T., Barros, B. D., Ye, C. D., Niu, X., Hoang, C., Tran, K., Hsu, B., Nadejde, M., Lakew, S., Mathur, P., Currey, A., and Federico, M. (2022). Sockeye 3: Fast neural machine translation with pytorch. *arXiv*, abs/2207.05851.
- Hunter, J. D. (2007). Matplotlib: A 2d graphics environment. *Computing in Science & Engineering*, 9(3):90–95.
- Khayrallah, H., Thompson, B., Duh, K., and Koehn, P. (2018). Regularized training objective for continued training for domain adaptation in neural machine translation. In Birch, A., Finch, A., Luong, T., Neubig, G., and Oda, Y., editors, *Proceedings of the 2nd Workshop on Neural Machine Translation and Generation*, pages 36–44, Melbourne, Australia. Association for Computational Linguistics.
- Kirkpatrick, J., Pascanu, R., Rabinowitz, N., Veness, J., Desjardins, G., Rusu, A. A., Milan, K., Quan, J., Ramalho, T., Grabska-Barwinska, A., Hassabis, D., Clopath, C., Kumaran, D., and Hadsell, R. (2017). Overcoming catastrophic forgetting in neural networks. *Proceedings of the National Academy of Sciences*, 114(13):3521–3526.
- Knowles, R. (2019). *Interactive and Adaptive Neural Machine Translation*. PhD thesis, The Johns Hopkins University, Baltimore, Maryland, USA.
- Knowles, R., Larkin, S., Tessier, M., and Simard, M. (2023). Terminology in neural machine translation: A case study of the Canadian Hansard. In Nurminen, M., Brenner, J., Koponen, M., Latomaa, S., Mikhailov, M., Schierl, F., Ranasinghe, T., Vanmassenhove, E., Vidal, S. A., Aranberri, N., Nunziatini, M., Escartín, C. P., Forcada, M., Popovic, M., Scarton, C., and Moniz, H., editors, *Proceedings of the 24th Annual Conference of the European Association for Machine Translation*, pages 481–488, Tampere, Finland. European Association for Machine Translation.

- Kothur, S. S. R., Knowles, R., and Koehn, P. (2018). Document-level adaptation for neural machine translation. In Birch, A., Finch, A., Luong, T., Neubig, G., and Oda, Y., editors, *Proceedings of the 2nd Workshop on Neural Machine Translation and Generation*, pages 64–73, Melbourne, Australia. Association for Computational Linguistics.
- Larkin, S., Joanis, E., Stewart, D., Simard, M., Foster, G., Ueffing, N., and Tikuisis, A. (2022). Portage Text Processing. <https://github.com/nrc-cnrc/PortageTextProcessing>.
- Li, Y., Zhao, L., Church, K., and Elhoseiny, M. (2020). Compositional language continual learning. In *8th International Conference on Learning Representations (ICLR)*.
- Lo, C.-k., Knowles, R., and Goutte, C. (2023). Beyond correlation: Making sense of the score differences of new MT evaluation metrics. In Utiyama, M. and Wang, R., editors, *Proceedings of Machine Translation Summit XIX, Vol. 1: Research Track*, pages 186–199, Macau SAR, China. Asia-Pacific Association for Machine Translation.
- Mathur, N., Baldwin, T., and Cohn, T. (2020). Tangled up in BLEU: Reevaluating the evaluation of automatic machine translation evaluation metrics. In Jurafsky, D., Chai, J., Schluter, N., and Tetreault, J., editors, *Proceedings of the 58th Annual Meeting of the Association for Computational Linguistics*, pages 4984–4997, Online. Association for Computational Linguistics.
- Moi, A. and Patry, N. (2022). Huggingface’s tokenizers. <https://github.com/huggingface/tokenizers>.
- Papineni, K., Roukos, S., Ward, T., and Zhu, W.-J. (2002). Bleu: a method for automatic evaluation of machine translation. In Isabelle, P., Charniak, E., and Lin, D., editors, *Proceedings of the 40th Annual Meeting of the Association for Computational Linguistics*, pages 311–318, Philadelphia, Pennsylvania, USA. Association for Computational Linguistics.
- Paszke, A., Gross, S., Massa, F., Lerer, A., Bradbury, J., Chanan, G., Killeen, T., Lin, Z., Gimelshein, N., Antiga, L., Desmaison, A., Kopf, A., Yang, E., DeVito, Z., Raison, M., Tejani, A., Chilamkurthy, S., Steiner, B., Fang, L., Bai, J., and Chintala, S. (2019). Pytorch: An imperative style, high-performance deep learning library. In *Advances in Neural Information Processing Systems 32*, pages 8024–8035. Curran Associates, Inc.
- Popović, M. (2015). chrF: character n-gram F-score for automatic MT evaluation. In Bojar, O., Chatterjee, R., Federmann, C., Haddow, B., Hokamp, C., Huck, M., Logacheva, V., and Pecina, P., editors, *Proceedings of the Tenth Workshop on Statistical Machine Translation*, pages 392–395, Lisbon, Portugal. Association for Computational Linguistics.
- Post, M. (2018). A call for clarity in reporting BLEU scores. In Bojar, O., Chatterjee, R., Federmann, C., Fishel, M., Graham, Y., Haddow, B., Huck, M., Yepes, A. J., Koehn, P., Monz, C., Negri, M., Névél, A., Neves, M., Post, M., Specia, L., Turchi, M., and Verspoor, K., editors, *Proceedings of the Third Conference on Machine Translation: Research Papers*, pages 186–191, Brussels, Belgium. Association for Computational Linguistics.
- Rei, R., C. de Souza, J. G., Alves, D., Zerva, C., Farinha, A. C., Glushkova, T., Lavie, A., Coheur, L., and Martins, A. F. T. (2022). COMET-22: Unbabel-IST 2022 submission for the metrics shared task. In Koehn, P., Barrault, L., Bojar, O., Bougares, F., Chatterjee, R., Costa-jussà, M. R., Federmann, C., Fishel, M., Fraser, A., Freitag, M., Graham, Y., Grundkiewicz, R., Guzman, P., Haddow, B., Huck, M., Jimeno Yepes, A., Kocmi, T., Martins, A., Morishita, M., Monz, C., Nagata, M., Nakazawa, T., Negri, M., Névél, A., Neves, M., Popel, M., Turchi, M., and Zampieri, M., editors, *Proceedings of the Seventh Conference on Machine Translation (WMT)*, pages 578–585, Abu Dhabi, United Arab Emirates (Hybrid). Association for Computational Linguistics.
- Rei, R., Stewart, C., Farinha, A. C., and Lavie, A. (2020). COMET: A neural framework for MT evaluation. In Webber, B., Cohn, T., He, Y., and Liu, Y., editors, *Proceedings of the 2020 Conference on Empirical Methods in Natural Language Processing (EMNLP)*, pages 2685–2702, Online. Association for Computational Linguistics.
- Wang, D., Wei, H., Zhang, Z., Huang, S., Xie, J., and Chen, J. (2022). Non-parametric online learning from human feedback for neural machine translation. *Proceedings of the AAAI Conference on Artificial Intelligence*, 36(10):11431–11439.
- Zhang, X., Shapiro, P., Kumar, G., McNamee, P., Carpuat, M., and Duh, K. (2019). Curriculum learning for domain adaptation in neural machine translation. In

Burstein, J., Doran, C., and Solorio, T., editors, *Proceedings of the 2019 Conference of the North American Chapter of the Association for Computational Linguistics: Human Language Technologies, Volume 1 (Long and Short Papers)*, pages 1903–1915, Minneapolis, Minnesota. Association for Computational Linguistics.

Álvaro Peris and Casacuberta, F. (2019). Online learning for effort reduction in interactive neural machine translation. *Computer Speech & Language*, 58:98–126.

A Data

Table 1 shows the sizes of the training, tuning, and test sets for our HP tuning and experiments; we provide some additional notes here.

In order to keep our setup as realistic as possible, we do not deduplicate the data (and the Hansard is known to have repetitive/boilerplate text). Regarding duplicates, within our 2000-sentence test and tuning sets, about 7.5% of sentences (either source or target) also appear in the training set. Looking at pairs of sentences (source and target) instead, this number falls to 4.75%. Internal repetition (sentences that repeat within a test or tuning set) is about 1%. Repeated sentences tend to be short: their average length is 10.3 words, compared to 20.1 words for all test and tuning sets.

B NMT Model Training

We train all models using Sockeye version 3.1.31 (Hieber et al., 2022), commit 13c63be5 with PyTorch 1.13.1 (Paszke et al., 2019). Table 3 lists the parameter settings in our experiments that differ from the Sockeye defaults. Training was performed on 4 Tesla V100-SXM2-32GB GPUs.

For data preprocessing, a bilingual SentencePieceUnigramTokenizer with a vocabulary of 32k tokens was trained using HuggingFace’s tokenizers (Moi and Patry, 2022) library version 0.14.1 on all of the 4498960 French and 4498960 English sentences from HP-BASELINE-INITIAL. The tokenizer also applies a few normalizations done by Portage (Larkin et al., 2022). Our vocabulary is augmented with generic tokens and other domain-specific tokens (unused in these experiments, but intended for future work on domain adaptation); this yields a final vocabulary of 32123 tokens.

Name	Value
amp	True
grading clipping type	abs
max sequence length	200:200
params	previous model when CL
batch size	8192
batch type	max-word
cache last best params	1
cache metric	bleu
cache strategy	last
checkpoint interval	10 ⁷
decode and evaluate	-1 (entire validation)
initial learning rate	see Table 4
keep last params	1
learning rate scheduler type	None
max epochs	see Table 4
metrics	perplexity & accuracy
min epochs	Same as max epochs
optimizer	Adam
optimizer Betas	0.9, 0.98
optimized metric	BLEU
update interval	2
vocabulary size	32121
attention heads	16:16
shared vocabulary	True
transformer FFN	4096:4096
transformer model size	1024:1024
weight tying	True

Table 3: Differences between Sockeye’s default parameters and our HP configuration.

B.1 Baselines

The HP-BASELINE-INITIAL was trained on the *HP-baseline-initial-train* data, with early stopping done using the *HP-baseline-initial-dev* set.

The HP-BASELINE-FINAL was trained using the same settings as the HP-BASELINE-INITIAL, using the relevant HP data described in Fig. 1 (i.e., early stopping based on *HP-baseline-final-dev* data).

To build an even stronger final BASELINE with a recency bias (i.e., recently exposed to all the HP CL data), we implemented an oracle finetuned model which we call HP-BASELINE-RECENCY. Using both the HP-BASELINE-INITIAL and HP-BASELINE-FINAL models as starting points, we finetuned them with the full set of CL data, using as the LR and number of epochs the full set used in the hyperparameter search (described in Section 5.2). We then selected the best finetuned model based on per-

formance on the *HP-epilogue* data (thus making this an oracle BASELINE). For EN-FR this was the one trained with $\text{LR}=3.0 \cdot 10^{-5}$ and 4 epochs starting from the HP-BASELINE-INITIAL model, while for FR-EN this was $\text{LR}=3.0 \cdot 10^{-6}$ and 64 epochs also starting from the HP-BASELINE-INITIAL model.

Using the full data sets, we built BASELINE models by applying the same approach as described for the HP-BASELINE models. All models were trained with the exact same configuration as for the HP experiments but using their respective corpora.

We also note differences in how we trained the BASELINE-RECENCY model for the full data. Using BASELINE-INITIAL’s configuration, we lowered its learning rate from 0.06325 to 0.006325 and initialized its weights with those of BASELINE-INITIAL, to finetune BASELINE-INITIAL on recent data. We use all stages except the last one for its training corpus. The last stage was used as a validation corpus to control early stopping.

B.2 Continual Learning

We start CL from the BASELINE-INITIAL model (either HP-BASELINE-INITIAL or BASELINE-INITIAL). For our realistic CL setup, we choose not to use a validation set and do not perform early stopping; that is, we set a fixed number of epochs (training iterations over the full stage of data) and train until those epochs have been completed. We also keep the learning rate fixed throughout the training, rather than using a variable learning rate. When the training on a stage is complete, we select the last saved model checkpoint (produced at the end of training) to use for translation of the test sets and to use as the starting model parameters for finetuning on the subsequent stage’s data. The choice not to use a validation set enables us to use the full stage data (which is already fairly small) for training. By doing this hyperparameter setting, we aim to pick a learning rate and number of epochs that are at low risk of overfitting to the data, while also providing desired improvements, thus aiming to achieve the same desired end goal as using a validation set. Also, this avoids the risk of using a validation set that is too small to draw accurate conclusions from.

¹¹We also examined this in the FR-EN direction and with chrF Popović (2015) implemented in sacre-bleu Post (2018) with signatures BLEU: nrefs: 1|case: mixed|eff: no|tok: 13a|smooth: exp|version: 2. 3. 2 and chrF: nrefs: 1|case: mixed|eff: yes|nc: 6|nw: 0|space: no|version: 2. 3. 2, but found similar results.

C Effects of Hyperparameters

Table 4 shows the set of learning rates and epochs used for our hyperparameter grid search.

Parameter	Values
initial learning rate	3e-4, 1e-4, 3e-5, 1e-5, 3e-6, 1e-6
max epochs	1, 2, 4, 8, 16, 32, 64, 128

Table 4: Grid search values for HP tuning.

In Fig. 5 (EN-FR), we can see how the combination of learning rate and number of epochs impacts performance.¹¹ The subplots show the minimum BLEU gain (with negative values indicating a degradation in BLEU), the median BLEU gain, and the maximum BLEU gain observed for a given set of hyperparameters as measured across stages 2-16 after training on stages 1-15, respectively.

The first notable result from these plots is that—as expected—we do see catastrophic forgetting if the learning rate, number of epochs, or both are too high. This is apparent in the lower right corner of all plots, where we see increasingly large drops in BLEU scores from the HP-BASELINE-INITIAL model to the CONTINUAL LEARNING models trained on various stages. A brief examination of a sample of the output for the high learning rate and large number of epochs suggests that the systems still retain the ability to generate output that is generally fluent, but that there are substitutions (likely due to overfitting on the previous stage) that can heavily impact adequacy.

The second notable result is that we find similar patterns between BLEU and chrF. Both are surface-level automatic evaluation metrics, so it is not a surprise to see this correlation, particularly for this well-studied language pair. Nevertheless, this replication of similar results across metrics can make us more confident that the observed patterns are real. This is particularly important because the BLEU and chrF differences are relatively small, and it is known that such small differences may not always correspond to perceivable significant differences if we were to perform human judgments (Mathur et al., 2020; Lo et al., 2023, i.a.). However, we do note that in this particular case, where the model may

Figure 5: EN-FR results for BLEU score gains from CL (over HP-BASELINE-INITIAL model) as measured over HP stages 2-16 (after finetuning on stages 1-15 respectively).

primarily be learning a new name or new term and otherwise leaving the output fairly similar to previous output, there may be reason to expect that these BLEU and chrF scores may capture genuine signal. We explored this question of volatility in Section 5.4 and the question of new terminology in Section 6.2.

Finally, we observe that there is a cluster of pairs of learning rates and number of epochs that show similar performance. Given the concerns about making decisions based on such small automatic score metric differences, we may instead choose to select from the pool of top systems on some other basis, such as training time required.

C.1 Selection of HPs

Our experimental setup is designed to let us explore the performance of different hyperparameter settings without any test set contamination. However, our main goal is to actually perform this CL in the longer-term, more realistic setting. To do this, we want to be able to select hyperparameters based on the results on the data used for our hyperparameter search; then we wish to see how these perform on the realistic data. We later run additional experiments on the full data for comparison, but we would like the initial run to be as realistic as possible: selecting just one set of parameters.

In order to do this, we need to decide on an approach to hyperparameter selection. We have a cluster of pairs of hyperparameters that perform similarly, with very small differences in automatic metric scores. For example, given a high-performing pair of hyperparameters, we may also find that a slightly larger number of epochs paired with a

slightly lower learning rate will perform similarly (or vice versa). So how should we choose between these? At first glance, we may wish to maximize the minimum of some automatic metric over the stages in our hyperparameter search; this, however, has the downside of overemphasizing the impact of a single stage (potentially problematic if one or more of the stages are unusual in some way that impacts automatic metric scores). Alternatively we may aim for consistency, selecting hyperparameters that exhibit a low standard deviation in automatic metric score differences; of course, this is no guarantee of strong performance, as a very low-performing system could also have a low standard deviation. And a high median on its own also fails to tell the whole story.

We are seeking to balance various interests in our selection of hyperparameters. These include plasticity (ability to learn from new data), stability (maintaining high performance on past data), low volatility (no dramatic changes from stage to stage), and cost (i.e., time, compute resources, or both). We also need to balance risk and reward, i.e., whether we want a model that reaches the highest scores but may also exhibit greater inconsistency or variance, or a model that may not obtain the highest automatic metric scores but is also fairly consistent in terms of overall performance.

As was evident from the heatmaps (Fig. 5), we have a number of different hyperparameter settings that perform quite similarly, clustered along the diagonal. When we examine their distributions over the hyperparameter tuning experiment stages via the

boxplots in Fig. 6, the large overlaps in interquartile range hammer home that these top performing hyperparameters are not significantly different from one another. Thus, for our initial experiment, we choose a high-performing (high median BLEU score, relatively low standard deviation) set of hyperparameters with a low number of epochs (in order to decrease training time and cost, particularly since not all users of these tools may have access to GPUs). For EN-FR this is $1.0 \cdot 10^{-5}$ LR and 8 epochs. For FR-EN this is $3.0 \cdot 10^{-6}$ LR and 8 epochs. We then also compare this against another HP setting. As noted earlier, there are a number of different ways one could choose between these, this heuristic is only one possible approach.

eventually begins to drop. This is not overly concerning in and of itself: as new data comes in, this may change preferred translations, potentially causing mismatches with the original reference data which would be viewed as improvements by the users on later data. With the lower learning rate we observe that, as expected, stability is improved and performance even increases slightly.

D.3 English–French COMET Figures

Figs. 10a and 10b use COMET (Rei et al., 2020) version `unbabel-comet==2.2.2` with model `Unbabel/wmt22-comet-da` to measure performance on the *epilogue-test*. They correspond to the BLEU score Figs. 3a and 3b, respectively.

Overall, we see similar trends between COMET and the BLEU scores used in the rest of the paper; if we had used COMET rather than BLEU for our hyperparameter selection, we still would have been choosing between a very similar set of top hyperparameters.

Figure 6: EN-FR boxplot showing BLEU score differences over stages with medians and interquartile range (left vertical axis) as well as standard deviation (red triangles and right vertical axis).

D Additional Figures

We provide additional figures in this section. Most scalable figures in this paper are produced using Matplotlib (Hunter, 2007), version 3.7.1.

D.1 French–English Figures

The results for FR-EN are similar to those from the lower learning rate for EN-FR, as we observe in Figs. 7 and 8.

D.2 Stability Figures

Fig. 9 shows the stability of CL models. Ideally, performance would remain relatively constant. With our initially selected learning rate performance

Figure 7: FR-EN incremental evaluation on the *epilogue-test* set after training on each stage of data, compared against BASELINES.

Figure 8: FR-EN stage-wise evaluation showing relative performance (Δ BLEU between CL model and BASELINE-INITIAL model) on individual CL stages.

(a) $1.0 \cdot 10^{-5}$ LR and 8 epochs

(b) $3.0 \cdot 10^{-6}$ LR and 8 epochs

Figure 9: EN-FR stability over CL stages, tested on held-out *baseline-initial-test* data.

(a) $1.0 \cdot 10^{-5}$ LR and 8 epochs

(b) $3.0 \cdot 10^{-6}$ LR and 8 epochs

Figure 10: EN-FR incremental evaluation using COMET on the *epilogue-test* set after training on each stage of data, compared against BASELINES, for two LR.