Retrieving Examples from Memory for Retrieval Augmented Neural Machine Translation: A Systematic Comparison

Maxime Bouthors[♡,♣], Josep Crego[♣], and François Yvon[♡]

[♡]Sorbonne Université, CNRS, ISIR, F-75 005 Paris, France
lastname@isir.upmc..fr

[♣]ChapsVision, 4 rue du Port aux Vins, F-92 150 Suresnes, France
{mbouthors,jcrego}@chapsvision.com

Abstract

Retrieval-Augmented Neural Machine Translation (RAMT) architectures retrieve examples from memory to guide the generation process. While most works in this trend explore new ways to exploit the retrieved examples, the upstream retrieval step is mostly unexplored. In this paper, we study the effect of varying retrieval methods for several translation architectures, to better understand the interplay between these two processes. We conduct experiments in two language pairs in a multi-domain setting and consider several downstream architectures based on a standard autoregressive model, an edit-based model, and a large language model with in-context learning. Our experiments show that the choice of the retrieval technique impacts the translation scores, with variance across architectures. We also discuss the effects of increasing the number and diversity of examples, which are mostly positive across the board.

1 Introduction

Retrieval-Augmented Language Models and Translation Models are getting a lot of traction (see (Li et al., 2022a) for a recent review). For translation tasks, the use of retrieval-based techniques that identify the most relevant segment(s) in a Translation Memory (TM) has long been used in professional Computer Aided Translation environments (Bowker, 2002), where the retrieved segments provide translators with valuable suggestions. Segments closely resembling the source sentence to be translated can also be directly edited to speed up translation. Such ideas have also been used in Machine Translation (MT), first in the example-based tradition (Nagao, 1984; Somers, 1999; Carl et al., 2004), then in the statistical-based paradigm (Koehn and Senellart, 2010), more recently for neural machine translation (NMT).

There are several ways to take advantage of translation examples in NMT architectures: Farajian et al. (2017) use a small set of examples to perform on-the-fly, lightweight, fine-tuning (using both source and target sides); Bulte and Tezcan (2019) simply concatenate the (target side of) a handful of examples on the source side of the encoder, leaving the rest of their autoregressive decoder unchanged; Xu et al. (2023) repurpose the edit-based architecture of Gu et al. (2019) to compute new translations from existing ones with a non-auto-regressive (NAT) decoder; finally, in-context learning (ICL) in large language models (LLMs) provides yet another way to seamlessly combine TMs with text generation (see Moslem et al. (2023), *inter alia*).

These studies (and several others, fully discussed in §5) not only differ in the way they use examples but also in the way TMs are searched, retrieved, and filtered. This makes the direct comparison between these proposals sometimes difficult to reproduce and analyze. Furthermore, it also prevents precisely assessing the computational complexity of the complete translation pipeline.

In this paper, we perform experiments with several representative retrieval methods that we systematically combine with multiple RAMT architectures. In doing so, our main goal is not to compare these downstream architectures, but rather to better understand the interplay between the retrieval and generation tasks, to make the trade-offs between these steps explicit, and to formulate recommendations regarding future uses of TMs in NMT.

Specifically, we address the following questions:

- How much does the example selection impact translation performance? Is one retrieval technique always better than the others, irrespective of the MT architecture, or is it necessary to adapt the former to the latter?
- Do we need multiple examples? If so, what makes a good set of examples? Does the diversity of examples help?

- By restricting retrieval to a restricted subset of examples based on the source domain, can we expect better performances, even though the quality of the best retrieved match is decreased?

We notably find that (a) retrieval actually matters for edit-based and in-context learning; (b) existing retrieval pipelines can be simplified at inference; (c) optimizing source coverage and/or instance diversity is helping, especially when the closest match is poor.

2 Retrieval in NMT architectures

2.1 Retrieval Pipeline

In the information retrieval (IR) framework, given a query q, a set of documents D is filtered and/or ranked according to a retrieval score. The challenge is to craft a score $s(q, d)$ that will retrieve documents d that are relevant for a downstream task. In MT, q is a source sentence, and D is a translation memory from which we can extract relevant (source, target) pairs ($d = (x, y)$). The retrieval process can be divided into three steps (Figure 1):

1. **Domain selection** selects the corpus to retrieve examples from, typically based on domain/genre similarity;

2. **Filtering** narrows down the set of relevant examples based on superficial comparison between the source query q and each example d.

    - Filtering can use a simple similarity score to filter TM candidates based on some minimal threshold: Bulte and Tezcan (2019) uses Jaccard similarity between bag-of-word representations; (Xu et al., 2020; Bouthors et al., 2023) use an n-gram match similarity;
    - Filtering can also be controlled by the specification of the indexing vocabulary, which typically excludes frequent words, thereby shortening the list of similar documents (see Appendix B.2).

3. **Ranking** uses a retrieval score such as n-gram overlap (Xu et al., 2020), BM25 score (Gu et al., 2018; Cheng et al., 2022), edit distance (ED) (Bulte and Tezcan, 2019; Xu et al., 2020; Bouthors et al., 2023), cosine similarity between q and x's embeddings (Xu et al., 2020; Pham et al., 2020; Vilar et al., 2023). *Incremental ranking* or *Weighted Coverage* can also be used to enforce diversity when retrieving multiple samples (Cheng et al., 2022; Agrawal et al., 2023; Sia and Duh, 2023a).

In a final **selection step**, only the top-k most similar candidates are eventually retained. Depending on the choice of the filtering parameters, the actual number of examples retrieved for a given query may be strictly smaller than k. For some domains, retrieval may even return an empty list.

2.2 Measuring Retrieval Quality

The effects of a retrieval strategy are only observed once a translation model is trained and evaluated. As this is an excessively costly process, we introduce several aspects that define *a priori* the quality of retrieved similar examples. For a single example d, this includes its semantic relatedness with q or the lexical overlap with the query; d's length also matters, as long examples may include irrelevant words that can hurt translation (Xu et al., 2020), and increase the computational processing cost. Now, looking at *sets of examples*, we would also like them to cover most query words, while remaining diverse and short on average.

To evaluate these facets, we compute the following scores for each set of similar examples:

- **Coverage** is the proportion of query tokens covered by the example tokens. It can be defined in several ways (bag-of-word recall, modified recall[1], n-way alignment score[2]).

- **Relevance** is the proportion of contributing tokens from the example tokens (with the same three underlying definitions as coverage). All other words are deemed lexically irrelevant; we report an average over examples.

- **Length** is the average number of tokens of retrieved examples.

In the next paragraphs, we describe how these quantities are controlled during retrieval.

2.3 Smoothed Longest Common Subsequence

The Levenshtein edit distance (LED) is widely used as the ranking function in RAMT (see §2.1).

[1]Each source token can only be covered at most once.

[2]Based on an alignment graph between examples and query that forbids swapping, as defined by Bouthors et al. (2023).

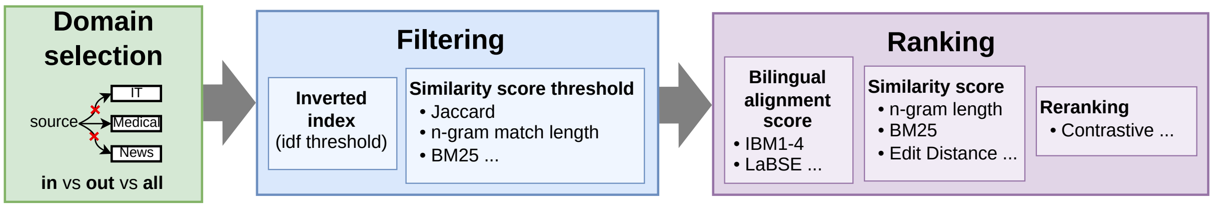

Figure 1: High-level overview of the retrieval pipeline in fuzzy-matching.

It counts the minimum number $\Delta(x, q)$ of word-level operations (deletion, insertion, replacement) required to edit x into q and then normalizes:

$$\text{LED}(x, q) = 1 - \frac{\Delta(x, q)}{\max(|x|, |q|)}, \tag{1}$$

Using LED mainly selects examples that are lexically similar to the source. As it penalizes non-matching parts (in the normalizer), it may discard long examples containing good matches in a substring. Such examples can still be relevant if they yield a high coverage of the query.

As an alternative to LED, setting the deletion cost to zero in (1) computes the Longest Common Subsequence (LCS) between x and q, which maximizes the coverage at the expense of relevance. This also means that there is no penalization for length, which can lead to long and hard-to-exploit examples. We propose a smoothed version, namely δ-LCS, with a small non-zero deletion cost δ. δ-LCS thus performs a trade-off between coverage, relevance, and length. Details are in Appendix A.

2.4 Controlling Diversity: Contrastive Retrieval

As is well known in the IR literature, it is unproductive to retrieve multiple identical examples. Diversity can help increase coverage without hurting the relevance of individual examples. For RAMT, a small number of diversity preserving approaches have been proposed: Cheng et al. (2022) (Levenshtein distance), Agrawal et al. (2023) (n-gram overlap) and Sia and Duh (2023b) (BM25) rely on an iterative algorithm inspired by the Maximal Marginal Relevance (MMR) criterion of Goldstein and Carbonell (1998). In a nutshell, this means that the ranking scoring function is iteratively updated to downgrade candidate examples that are either too similar to already selected examples or that cover already covered words. In our experiments, we follow Cheng et al. (2022), and after selecting $|M|$ matches in M, we penalize the ranking scores of remaining candidates with the following term:

$$\frac{\alpha}{|M|} \sum_{x \in M} \text{LED}(\cdot, x), \tag{2}$$

where $\alpha > 0$ controls the strength of the penalty.

2.5 Integrating TMs in Translation

In our comparisons, we consider three NMT architectures with variants. The first, called Neural Fuzzy Augmentation (NFA), implements the autoregressive approach of Bulte and Tezcan (2019), with minor variants. The second, `TM-LevT`, is edit-based and mostly follows Xu et al. (2023), as recently extended by Bouthors et al. (2023) to handle multiple matches. The third is based on in-context learning (ICL) with large LMs, using the causal BLOOM LM (BigScience et al., 2022) and the HuggingFace Transformer library[3] to run the experiments. We provide full details regarding these architectures in Appendix C. At a high level, the most important distinction is between the *autoregressive generative approaches* of NFA and ICL, which both use an enriched context comprising the source sentence and additional source and/or target matches, and `TM-LevT`, which tries to reuse, via an editing process, subparts of the retrieved matches. Another major difference is between ICL, which inputs the source side of matches for decoding, and the other two approaches, which do not need it during generation. This notably impacts the computational cost of encoding the context.

3 Data and Metrics

3.1 Data

We consider two translation directions, from English to French (en-fr) and from German to English (de-en), and experiment with multiple domains. This allows us to study a wide range of settings, with a varying density of matches: our datasets include ECB, EMEA, Europarl, GNOME, JRC-Acquis, KDE4, PHP, Ubuntu, OpenSubtitles,

[3]`https://github.com/huggingface/transformers`.

and Koran[4] (statistics in Tables 1 and 2). For en-fr, these datasets reproduce the setting of (Xu et al., 2020). [5] For de-en, we reuse the data prepared by Koehn and Knowles (2017) with the split of Aharoni and Goldberg (2020).[6] The most favorable situation is to translate in a narrow domain with large TMs, ensuring that multiple good matches can be found (e.g. JRC-Acquis and EMEA). However, in a narrow domain, the TM can sometimes be small (e.g., Ubuntu), and this is where other related domains can also help. On the other end of this spectrum, Europarl or News-Commentary are thematically very diverse and good matches much harder to find.

For each dataset $\mathcal{D}$, we compute a *density* score based on the number of connected components (NCC) in a similarity graph Γ. Two translation examples are linked in Γ if their similarity score (1) is greater than 0.4:

$$\text{density}(\mathcal{D}) = 1 - \frac{1 - \text{NCC}(\Gamma)}{1 - |\mathcal{D}|} \qquad (3)$$

In high-density domains, it is thus easier to retrieve relevant translation examples (see Tables 1 and 2).

Note that these data are not ideal. First, for some domains, the corresponding data may be included in the very large corpora used to train LLMs. In our experiments with BLOOM, which is trained on the ROOTS corpus (Laurençon et al., 2022), this is the case for JRC-Acquis, Wikipedia, Europarl, TEDTalks (en-fr).[7]

Furthermore, the en-fr test sets have been selected based on the existence of at least one close example in the same domain, using the standard LED to compute similarities. More precisely, the 1,000 instances in *test-0.6* always have at least one match with similarity greater than 0.6, for *test-0.4* the nearest match has a similarity comprised between 0.4 and 0.6 (details in (Xu et al., 2020)).[8]

This design allows us to focus on the effect of retrieval quality (medium vs high-scoring matches) on translation scores. It however yields absolute scores that do not compare with what would be obtained with a fully randomized selection process. For a more realistic evaluation, we use the de-en data, which, however, cover fewer domains. In general, our experiments are more thourough with en-fr data as this language pair was used to select a subset of interesting configurations to be then also tested for de-en.

As a last word of caution, we observe that some domains are much easier to translate than others: JRC-Acquis is very repetitive, which yields BLEU scores in the high 70's; NewsCommentary, on the other hand barely achieves BLEU scores higher than 20. Averaged results should be looked at with care - only per-domain scores can tell the full story (Appendix E).

3.2 Metrics

We report BLEU scores (Papineni et al., 2002) computed by SacreBLEU (Post, 2018),[9] as well as COMET-22[10] scores (Rei et al., 2022) using the official implementation. Additionally, we use the multi-reference sentence BLEU scores between the target side of examples, and the translation output[11], averaged over corpora, to evaluate the *copy rate* of systems, i.e. their ability to recopy subparts of the retrieved examples.

3.3 Implementation and Parameters

We use in-house, open-source[12] libraries that implement the various retrieval methods explored in this paper. Details regarding parameter settings are in Appendix B. For translation architectures, refer to Appendix C.

In our experiments, we contrast three strategies for domain selection: in-domain, out-of-domain, no-selection. Regarding filtering (step 2 in Figure 1), we compare n-gram matching (NGM), BM25, and no filter. NGM filters out examples unless they share a common n-gram g with the source q of relative length greater than a threshold τ (e.g. $\frac{|g|}{|q|} \geq \tau$). As for BM25, we only retain the L best BM25 candidates.

Finally, regarding ranking, we compare various definitions of the edit distance (ED) (see §2.3) with BM25.

Computational issues In our experiments below, we mostly analyze retrieval results. However, note that each retrieval pipeline yields specific training

[4]These data can be downloaded from the OPUS website (`https://www.opus.nlpl.eu`) (Tiedemann, 2012).

[5]Splits from `https://github.com/jitao-xu/tm-levt`.

[6]This is the *test-de* test set.

[7]For these domains, the ICL scores have been disregarded.

[8]As we use our own reimplementation of edit distances, we have observed rare cases where these conditions were not exactly met.

[9]signature: `nrefs:1|case:mixed|eff:no|tok:13a|smooth:exp|version:2.1.0;`

[10]`Unbabel/wmt22-comet-da`

[11]Brevity penalty (BP) is removed.

[12]https://github.com/SYSTRAN/fuzzy-match

domain	ECB	EME	Epp	GNO	JRC	KDE	News	PHP	TED	Ubu	Wiki	**all**
size	195k	373k	2.0M	55k	503k	180k	151k	16k	159k	9k	803k	4.4M
avg length	29.2	16.7	26.6	9.4	28.8	10.5	26.4	14.5	17.7	5.2	19.6	18.6
density %	87.1	96.9	49.3	85.96	86.76	84.18	11.60	63.59	53.78	16.89	55.25	62.8

Table 1: Number of samples, average length of tokenized sentences, density for training sets (en-fr).

domain	it	kor	law	med	sub	**all**
size	223k	18k	467k	248k	500k	1.46M
mean length	9.6	20.4	28.0	15.5	8.1	16.3
density %	53.2	51.4	58.6	69.5	74.2	61.4

Table 2: Number of samples, average length of tokenized sentences, density for training sets (de-en).

and inference computational costs. Domain selection always speeds up the subsequent steps, with a very strong impact when the target domains are small. Regarding filtering, NGM has an algorithmic complexity $O(\bar{\ell}\log(n\bar{\ell}))$ for a single query using suffix array – with $\bar{\ell}$ the average sentence length and n the TM size – whereas BM25's complexity is $O(n|q|)$. Finally, regarding ranking, ED calculation takes $(O(n\bar{\ell}|q|))$, so again, linear w.r.t. n for one query.

4 Experiments

4.1 Comparing Retrieval Techniques

We first measure how much a change in the retrieval technique actually affects the instances that are eventually retrieved. For this, we compute bag-of-word coverage, relevance, and length (introduced in §2.2). We compare a baseline NGM filter using the LED ranker, as used in Bouthors et al. (2023), with filter-free pipelines. The corresponding results for all testsets are in Table 3. LED yields the highest relevance, while δ-LCS and contrastive ranking yield a higher coverage. Overall, changing the retrieval technique does impact the set of instances that are used in training and inference.

4.2 Interactions Between Retrieval and Translation

In this section, we look at the interactions between retrieval and translation and systematically vary the retrieval component for the three architectures of §2.5. Notably, we compare two filters (NGM and BM25) during training and also contrast with a filter-free version in inference. We also vary the edit costs and the number of retrieved examples.

4.2.1 Architectures Comparison

Neural fuzzy augmentation We observed, in preliminary results, that NFA is insensitive to the retrieval setting at training time. We only report results for a model trained on the baseline setting NGM+LED ($\tau = 0.3$), then used in inference in a filter-free setting.

ranker	*test-0.4*	*test-0.6*	*test-de*
NGM+LED 1-1	55.1	64.3	-
NGM+LED 3-1	54.8	63.9	-
NGM+LED 3-2	54.8	64.2	-
NGM+LED 3-3	54.9/44.6	64.3/45.7	41.6
BM25	54.7/44.6	64.2/45.6	-
BM25$_c$	54.7/44.6	64.2/45.6	-
LED	54.9/44.7	64.4/45.7	41.7
δ-LCS	54.8/44.6	64.3/45.7	41.9
δ-LCS$_c$	54.8/44.6	64.3/45.7	41.8

Table 4: Average BLEU (/COMET ($\times$100)) scores for en-fr (11 domains) and de-en (5 domains) using NFA models. k_t-k_i in NGM+LED denotes a model trained with k_t examples, while inference uses k_i; $_c$ denotes contrastive ranking.

Results in Table 4 are very consistent and hardly vary across domains (see Table 13 in Appendix E) and language pairs. This is a first important result that somehow consolidates observations already performed for this model, which seems to be robust with respect to variations in the retrieval strategy.

Edit-based techniques Regarding edit-based approaches, we train Multi-Levenshtein Transformer (`TM`^{3}`-LevT`) on the same dataset, comparing a set of retrieval settings and both NGM and BM25 filters. We report the following results in Table 5:

- The setting used in the original `TM`^{3}`-LevT` paper: NGM+LED ($\tau = 0.3$) both at training and inference time.
- The best-performing train and inference setting pairs, as identifiable in Appendix D, for NGM and BM25 filters separately.
- We evaluate our best overall training pipeline (BM25+LED) on a filter-free setup with varying ED costs (using $\delta = 0.1$).

filter	NGM ($\tau = 0.3$)	-	-	-	-	-	-
ranker	LED	LED	LED	δ-LCS	δ-LCS	BM25	BM25
Contrast	-	-	$\alpha = 0.3$	-	$\alpha = 0.3$	-	$\alpha = 0.3$
coverage	48.1 65.6 38.1	58.2 68.9 57.0	60.8 70.5 -	63.8 71.9 62.5	**65.8** **73.5** **64.4**	62.2 70.0 -	62.3 70.0 -
relevance	40.3 **53.5** 31.4	**44.7** **53.5** **43.7**	44.1 **53.3** -	39.8 49.4 40.1	36.6 44.7 36.6	42.4 48.9 -	42.4 48.9 -
length	15.7 15.2 15.8	16.4 15.4 16.9	16.5 15.4 -	24.8 19.7 26.9	27.0 21.8 29.0	20.1 19.0 -	20.1 19.0 -

Table 3: Retrieval scores averaged over domains for triplets *test-0.4*, *test-0.6*, *test-de*. $\delta = 0.1$

filter+ranker	*test-0.4*	*test-0.6*	*test-de*
NGM+LED	43.9/26.2	56.0/47.8	29.4
best NGM+ED pair	45.5/**32.8**	56.9/49.7	-
best BM25+ED pair	45.7/31.0	57.1/49.9	-
LED (k=1)	44.9/29.0	55.7/46.6	-
LED (k=2)	45.2/29.1	56.7/48.5	-
LED	45.6/31.3	57.3/**50.7**	33.4
δ-LCS	45.9/31.4	**57.4**/50.5	**33.9**
δ-LCS$_c$	**46.0**/31.3	57.2/49.4	33.7

Table 5: Average BLEU (/COMET $(\times 100)$) scores across all 11 (en-fr) or 5 (de-en) domains using `TM`^{3}`-LevT`.

ranker	k-shot	*test-0.4*	*test-0.6*	*test-de*
LED	1	45.8/30.5	50.0/30.7	-
LED	3	47.8/35.3	51.5/35.1	33.3
LED$_c$	3	**48.3/37.5**	**52.0/37.3**	-
δ-LCS	3	48.1/36.0	51.6/36.3	33.7
δ-LCS$_c$	3	48.2/36.4	51.7/36.8	**33.9**
BM25	1	45.8/28.7	49.6/27.1	-
BM25	3	48.1/35.0	51.8/34.8	-
BM25$_c$	3	48.1/34.8	51.4/35.2	-

Table 6: Average BLEU/COMET $(\times 100)$ scores averaged accross **7** en-fr domains (*test-0.4/6*) and 5 de-en domains (*test-de*) for ICL (k-shot) for several filter-free retrieval setups. $_c$ denotes contrast; $\delta = 0.1$.

We observe here a stronger impact of retrieval on the downstream scores, with a large gain over the baseline (for test-0.4 and en-de). 0.1-LCS slightly outperforms LED in most conditions and metrics, with a very small difference between the contrastive and non-contrastive versions of the ranking.

In-context learning We evaluate BLOOM in k-shot translation for $k = 1$ and 3 with two rankers: ED and BM25.[13] The retrieval scope is always "in-domain". We do not use any filter to ensure the retrieval of exactly k examples for each test instance.[14] Results in Table 6 show again small differences between retrieval techniques, with a positive effect of contrastive ranking policy, which yields the best results. For this architecture, we also note that retrieving at least very good match (e.g. *test-0.6*) does not necessarily imply very high BLEU scores, contrarily to NFA and `TM-LevT`.

4.3 Complementary Analyses

What makes a good set of examples? We use a linear model and try to predict BLEU scores using the retrieval metrics (coverage, relevance, and length) of §2.2 for `TM`^{3}`-LevT` and ICL w.r.t. coverage, relevance, and length. First, with `TM`^{3}`-LevT`, the linear model has an average squared residuals of 0.2 BLEU. On the other hand, a constant model, which supposes that the model-agnostic metrics are independent of BLEU, has an error of 1.2. Coverage, relevance, and length have respective coefficients of 0.13, 0.09, and 0.03. This highlights the importance of coverage and relevance measures in the explanation of BLEU performance. As for ICL, the constant model has the same average residual error (0.28 against 0.26), with respective coefficients of 0.04, 0.03, and 0.00. Thus, ICL seems more robust to changes in the retrieved examples.

Copying input tokens The *copy rate*, introduced in §3.2 measures how much the translation model exploits slices from the examples to produce its output. Pipelines with high covering examples systematically imply a higher *copy rate*. We find that *copy rate* is correlated with higher BLEU scores for `TM`^{3}`-LevT` and ICL; in contrast NFA fails to produce higher BLEU scores with increasing *copy rate*.

Also note that, even though it relies on explicit edits, `TM`^{3}`-LevT` always has a lower *copy rate* than NFA or ICL; the latter notably has the highest *copy rate* for *test-0.6* and also the worst translation scores, suggesting that too many irrelevant tokens are kept in the output.

Domain Filtering Relaxing the constraint that similar examples should be retrieved "in-domain" increases the retrieval rate. However, it turns out to be detrimental for all architectures: results are in Table 8, where we compare in-domain retrieval

[13] We also consider a "random" ranking policy for contrast, which yields comparatively very poor results that are about 15 BLEU points below the others for $k = 3$. This confirms the findings of Moslem et al. (2023, Table 1, p. 235) on the benefits of TM-based retrieval.

[14] In some situations however, we could not find 3 candidates with a score greater than 0.

		test-0.4	*test-0.6*	*test-de*
NFA				
BM25		47.0	62.1	-
BM25$_c$		46.3	61.3	-
LED		43.4	60.8	38.2
δ-LCS		46.7	62.8	41.0
δ-LCS$_c$		**47.5**	**63.8**	**41.3**
TM^3-LevT				
NGM+LED		37.4	56.3	30.4
best BM25+ED pair		42.4	57.8	-
LED		40.7	56.9	37.2
δ-LCS		42.8	58.3	**39.5**
δ-LCS$_c$		**43.0**	**58.4**	39.4
ICL	k-shot			
LED	1	40.5	53.5	-
LED	3	51.8	64.7	45.0
LED$_c$	3	53.7	65.9	-
δ-LCS	3	54.1	66.1	47.2
δ-LCS$_c$	3	54.3	65.2	**48.1**
BM25	1	53.9	66.1	-
BM25	3	54.1	66.1	-
BM25$_c$	3	**54.9**	**67.0**	-

Table 7: Average *copy rate* of all three models.

with all-domains and out-of-domain retrievals.

domain	NFA	TM^3-LevT	ICL
In	54.9 / 64.4	45.6 / 57.3	47.8 / 51.5
All	52.5 / 62.3	43.8 / 55.4	45.1 / 49.0
Out	45.5 / 51.2	30.2 / 33.5	30.7 / 29.6

Table 8: Average BLEU scores (*test-0.4* / *test-0.6*, en-fr) according to the domain selection strategy, using filter-free LED as ranker.

The impact of the "all-domain" policy is genuine: when this policy is enforced, the most similar examples are found out-of-domain for 35.1% of our 22k test samples. The most impacted domains are Ubuntu (82.8% matches are out-of-domain) and NewsCommentary (79.7%). The per-domain analysis (Appendix E) however shows that this is detrimental for Ubuntu (-6.1/-6.7 BLEU for *test-0.4*/*test-0.6*), and neutral for NewsCommentary (-0.2/+0.2 BLEU). As expected, enforcing an "out-of-domain" selection constraint yields dramatic losses in BLEU (-15.4/-23.8 BLEU).

These results confirm the benefit of retrieving "in-domain", even for small domains: not only does it greatly speed up retrieval, but it also yields better examples and, ultimately, higher translation scores.

Simplifying the Retrieval Pipeline For large domains, removing the filtering step in the retrieval pipeline considerably increases the computational cost, especially during training.[15] Yet, it may prematurely discard useful examples, especially when using contrastive ranking. To evaluate this, we turn off filtering for test samples during inference. This simplification of the pipeline improves the scores for TM^3-LevT, while there is no effect for NFA (see lines for filtering free inference in Tables 4 and 5). Thus, for the former method at least, a trade-off can be made between latency and translation scores.

[15]The complexity is quadratic w.r.t. to the TM size.

Increasing the number of examples We vary k, the number of TM examples retrieved, from 1 to 3. Overall, we observe a gain (BLEU/COMET) when k increases. For ICL, this is already clear from the results in Table 6 where 3-shots clearly outperforms 1-shot. We get a similar conclusion for TM^3-LevT based on the results in Table 5, where we vary the inference procedure for a model trained on (BM25+LED). The test retrieval is filter-free LED with either exactly $k = 1$, 2, or 3 retrieved examples.

As for NFA, a model trained on up to 3 instances slightly benefits from more examples but does not compete with a model using only the one-best match in training and inference. This seems to contradict Bulte and Tezcan (2019), who claim the superiority of using more examples.

Optimizing for coverage with δ-LCS By design, δ-LCS retrieves examples having a higher coverage of the source than LED, which turns into higher *copy rates* for all architectures. For ICL and TM^3-LevT, it yields similar BLEU gain (ICL: +0.3/+0.1/+0.4; TM^3-LevT: +0.3/+0.1/+0.5 on resp. *test-0.4*, *test-0.6*, *test-de*). The analysis in Appendix D shows a consistent benefit of δ-LCS for TM^3-LevT when coupled with filters at training (NGM) and inference time (NGM and BM25).

Enforcing diversity with contrastive ranking We observe that using a contrastive ranker is mostly beneficial for medium-scoring similar examples (*test-0.4*), regardless of the architecture. It can even be detrimental when at least one high-matching example is found. This is because contrastive ranking generates less similar examples that are not necessarily relevant. In comparison, for *test-0.4*, increasing the diversity in retrieval seems beneficial, as it increases the coverage of the source. This suggests that contrastive methods should adapt their strength parameter (α in (2)) to the retrieval scores, enforcing more diversity when matches are poor.

5 Related Work

As for other text generation applications (Li et al., 2022b), efforts to integrate a retrieval component in NMT have intensified in recent years. One motivation is to increase the transparency of ML models by providing users with the retrieved examples that were used to compute the actual output (Rudin, 2019). For MT, this is achieved by integrating fuzzy matches retrieved from memory as an additional context. This can be performed by concatenating the retrieved target instance to the source text, an approach that also accommodates several TM matches (Bulte and Tezcan, 2019), or by the simultaneous use of their source and target sides (Pham et al., 2020; Reheman et al., 2023). More complex schemes to combine retrieved examples with the source sentence are in (Gu et al., 2018; Xia et al., 2019; He et al., 2021b). The recent studies of Cheng et al. (2022); Agrawal et al. (2023) and Sia and Duh (2023b) handle several complementary TM examples retrieved in a *contrastive manner* that aims to enhance source coverage. Gupta et al. (2023) propose a general formulation in the application of ICL in various tasks. Cai et al. (2021) also handle multiple matches and introduce two novelties: (a) retrieval is performed directly in the target language and (b) similarity scores are trainable, which allows to evaluate retrieved instances based on their usefulness in translation. Most of these attempts rely on auto-regressive (AR) decoding, meaning that the impact of TM match(es) on the output is only indirect.

The use of TM memory match with a NAT decoder is studied by Niwa et al. (2022); Xu et al. (2023); Zheng et al. (2023), who adapt `LevT` for this specific setting, using one single retrieved instance to initialize the edit-based decoder; (Bouthors et al., 2023) extends this technique to process multiple retrieved examples. Zhang et al. (2018) explore a different set of techniques to improve translation using retrieved segments instead of full sentences. Generalizing nearest neighbor language models (NNLMs) (He et al., 2021a) to conditional LMs, Khandelwal et al. (2021) perform k-NNMT as follows: at each decoding step, the k target contexts that closest to the current contextualized representations are retrieved and used to select the next token. This approach is further elaborated in (Zheng et al., 2021; Meng et al., 2022) and extended to chunks by Martins et al. (2022).

A final thread of relevant papers concerns the use of large language models, which, provided with suitable prompts and in-context examples, can be turned into effective translation systems. Such approaches have been tested with most LLMs, with the goal to illustrate the multi-tasking abilities of such models. Closer to our work, a series of work have tried to optimize LLMs performance for the MT task, systematically studying the effect of the prompt change, of the number of shots, and of the in-context examples selection procedures (Vilar et al., 2023; Zhang et al., 2023; Hendy et al., 2023; Bawden and Yvon, 2023). Moslem et al. (2023) were the first to combine LLMs with TMs, using an embedding-based retrieval system and combining (via concatenation) up to 5 TM-matches in the MT prompt; (Mu et al., 2023) followed suit, with a different LLM and a two-stage retrieval strategy (first 500 closest matches for a Lucene-based engine; then using up to 9 closest matches for the edit-distance). (Agrawal et al., 2023) studies a way to optimally select k examples so as to maximize coverage, an approach akin to our "contrastive” scenario – using a BM25 retriever in a first stage, and a greedy heuristic selection in a second stage. (Sia and Duh, 2023b) explores another benefit of selecting good in-context examples, that of maintaining consistency in the generated text - for this, they retrieve examples from a moving context window of past translations. Finally, M et al. (2023) go one step further by training a linear regression model predicting the goodness of TM instances based on a small set of features.

6 Conclusion

This paper has investigated the effect of varying the retrieval strategy for three commonly used retrieval-augmented machine translation architectures, trying to get a better understanding of the interplay of these two components. While auto-regressive encoder-decoder architecture seems quite robust w.r.t. changes in the retrieval strategy, this is less so for the two other architectures, for which optimizing the retrieval policy can yield significant returns. Our experiments have also highlighted the benefits of coverage-oriented retrieval policies, based on LCS, especially for the non-autoregressive model. Finally, we have validated the use of the “in-domain” selection policy and proposed to simplify the inference step by eliminating the filtering process, yielding better performance at the expense of an increased latency.

In our future work, we would like to continue the exploration of the interplay between retrieval and translation, with the aim to jointly optimize these two processes rather than have them designed independently.

7 Limitations

In this paper, we have focused on purely transductive techniques, meaning that all inference is performed with a frozen network - this limitation certainly needs to be reconsidered, and our results would be stronger with additional comparisons with e.g., on-the-fly fine-tuning (Farajian et al., 2017) or low-rank adaptation techniques (Hu et al., 2022).

We have chosen to use only one large LLM, with 176b million parameters. This was motivated by (a) the openness of the model and the transparency of the training data, which allowed us to control for test samples occurring also in the training; (b) the existence of multiple previous experiments with this model, which allowed us to get a reasonable idea of its basic translation abilities. More recent, smaller, and arguably better models (e.g., the LLAMA (Touvron et al., 2023) and Falcon families) (Almazrouei et al., 2023) with various levels of multilingual support (Alves et al., 2024), would likely yield a more faithful picture of the current performance of in-context learning with LLMs.

Our discussion has focused on measures of translation quality; in practical applications, computational costs associated with a specific combination of retrieval and architecture also matter. While we have tried to be explicit about the complexity of each retrieval algorithm, we have left aside issues related to identifying the optimal computation/performance tradeoffs.

8 Acknowledgements

This work was performed using HPC/AI resources from GENCI-IDRIS (Grants 2022-AD011013583 and 2023-AD010614012). The third author was also partly funded by the French National Research Agency (ANR) under grant ANR-23-IAS1-0006 (TraLaLam).

References

Sweta Agrawal, Chunting Zhou, Mike Lewis, Luke Zettlemoyer, and Marjan Ghazvininejad. 2023. In-context examples selection for machine translation. In *Findings of the Association for Computational Linguistics: ACL 2023*, pages 8857–8873, Toronto, Canada. Association for Computational Linguistics.

Roee Aharoni and Yoav Goldberg. 2020. Unsupervised domain clusters in pretrained language models. In *Proceedings of the 58th Annual Meeting of the Association for Computational Linguistics*, pages 7747–7763, Online. Association for Computational Linguistics.

Ebtesam Almazrouei, Hamza Alobeidli, Abdulaziz Alshamsi, Alessandro Cappelli, Ruxandra Cojocaru, Mérouane Debbah, Étienne Goffinet, Daniel Hesslow, Julien Launay, Quentin Malartic, Daniele Mazzotta, Badreddine Noune, Baptiste Pannier, and Guilherme Penedo. 2023. The falcon series of open language models.

Duarte M. Alves, José Pombal, Nuno M. Guerreiro, Pedro H. Martins, João Alves, Amin Farajian, Ben Peters, Ricardo Rei, Patrick Fernandes, Sweta Agrawal, Pierre Colombo, José G. C. de Souza, and André F. T. Martins. 2024. Tower: An open multilingual large language model for translation-related tasks.

Rachel Bawden and François Yvon. 2023. Investigating the translation performance of a large multilingual language model: the case of BLOOM. In *Proceedings of the 24th Annual Conference of the European Association for Machine Translation*, pages 157–170, Tampere, Finland. European Association for Machine Translation.

Workshop BigScience, :, Teven Le Scao, Angela Fan, Christopher Akiki, Ellie Pavlick, Suzana Ilić, Daniel Hesslow, Roman Castagné, Alexandra Sasha Luccioni, François Yvon, Matthias Gallé, Jonathan Tow, Alexander M. Rush, Stella Biderman, Albert Webson, Pawan Sasanka Ammanamanchi, Thomas Wang, Benoît Sagot, Niklas Muennighoff, Albert Villanova del Moral, Olatunji Ruwase, Rachel Bawden, Stas Bekman, Angelina McMillan-Major, Iz Beltagy, Huu Nguyen, Lucile Saulnier, Samson Tan, Pedro Ortiz Suarez, Victor Sanh, Hugo Laurençon, Yacine Jernite, Julien Launay, Margaret Mitchell, Colin Raffel, Aaron Gokaslan, Adi Simhi, Aitor Soroa, Alham Fikri Aji, Amit Alfassy, Anna Rogers, Ariel Kreisberg Nitzav, Canwen Xu, Chenghao Mou, Chris Emezue, Christopher Klamm, Colin Leong, Daniel van Strien, David Ifeoluwa Adelani, Dragomir Radev, Eduardo González Ponferrada, Efrat Levkovizh, Ethan Kim, Eyal Bar Natan, Francesco De Toni, Gérard Dupont, Germán Kruszewski, Giada Pistilli, Hady Elsahar, Hamza Benyamina, Hieu Tran, Ian Yu, Idris Abdulmumin, Isaac Johnson, Itziar Gonzalez-Dios, Javier de la Rosa, Jenny Chim, Jesse Dodge, Jian Zhu, Jonathan Chang, Jörg Frohberg, Joseph Tobing, Joydeep Bhattacharjee, Khalid Almubarak, Kimbo Chen, Kyle Lo, Leandro Von Werra, Leon Weber, Long Phan, Loubna Ben allal, Ludovic Tanguy, Manan Dey, Manuel Romero Muñoz, Maraim Masoud, María Grandury, Mario Šaško, Max Huang, Maximin Coavoux, Mayank Singh, Mike Tian-Jian Jiang, Minh Chien Vu, Mohammad A. Jauhar, Mustafa Ghaleb, Nishant Subramani, Nora Kassner, Nurulaqilla Khamis, Olivier Nguyen, Omar Espejel, Ona de Gibert, Paulo Villegas, Peter Henderson, Pierre Colombo, Priscilla Amuok, Quentin Lhoest, Rheza Harliman, Rishi Bommasani, Roberto Luis López, Rui Ribeiro, Salomey Osei, Sampo Pyysalo, Sebastian Nagel, Shamik Bose, Shamsuddeen Hassan Muhammad, Shanya Sharma, Shayne Longpre, Somaieh Nikpoor, Stanislav Silberberg, Suhas Pai, Sydney Zink, Tiago Timponi Torrent, Timo Schick, Tristan Thrush, Valentin Danchev, Vassilina Nikoulina, Veronika Laippala, Violette Lepercq, Vrinda Prabhu, Zaid Alyafeai, Zeerak Talat, Arun Raja, Benjamin Heinzerling, Chenglei Si, Davut Emre Taşar, Elizabeth Salesky, Sabrina J. Mielke, Wilson Y. Lee, Abheesht Sharma, Andrea Santilli, Antoine Chaffin, Arnaud Stiegler, Debajyoti Datta, Eliza Szczechla, Gunjan Chhablani, Han Wang, Harshit Pandey, Hendrik Strobelt, Jason Alan Fries, Jos Rozen, Leo Gao, Lintang Sutawika, M Saiful Bari, Maged S. Al-shaibani, Matteo Manica, Nihal Nayak, Ryan Teehan, Samuel Albanie, Sheng Shen, Srulik Ben-David, Stephen H. Bach, Taewoon Kim, Tali Bers, Thibault Fevry, Trishala Neeraj, Urmish Thakker, Vikas Raunak, Xiangru Tang, Zheng-Xin Yong, Zhiqing Sun, Shaked Brody, Yallow Uri, Hadar Tojarieh, Adam Roberts, Hyung Won Chung, Jaesung Tae, Jason Phang, Ofir Press, Conglong Li, Deepak Narayanan, Hatim Bourfoune, Jared Casper, Jeff Rasley, Max Ryabinin, Mayank Mishra, Minjia Zhang, Mohammad Shoeybi, Myriam Peyrounette, Nicolas Patry, Nouamane Tazi, Omar Sanseviero, Patrick von Platen, Pierre Cornette, Pierre François Lavallée, Rémi Lacroix, Samyam Rajbhandari, Sanchit Gandhi, Shaden Smith, Stéphane Requena, Suraj Patil, Tim Dettmers, Ahmed Baruwa, Amanpreet Singh, Anastasia Cheveleva, Anne-Laure Ligozat, Arjun Subramonian, Aurélie Névéol, Charles Lovering, Dan Garrette, Deepak Tunuguntla, Ehud Reiter, Ekaterina Taktasheva, Ekaterina Voloshina, Eli Bogdanov, Genta Indra Winata, Hailey Schoelkopf, Jan-Christoph Kalo, Jekaterina Novikova, Jessica Zosa Forde, Jordan Clive, Jungo Kasai, Ken Kawamura, Liam Hazan, Marine Carpuat, Miruna Clinciu, Najoung Kim, Newton Cheng, Oleg Serikov, Omer Antverg, Oskar van der Wal, Rui Zhang, Ruochen Zhang, Sebastian Gehrmann, Shachar Mirkin, Shani Pais, Tatiana Shavrina, Thomas Scialom, Tian Yun, Tomasz Limisiewicz, Verena Rieser, Vitaly Protasov, Vladislav Mikhailov, Yada Pruksachatkun, Yonatan Belinkov, Zachary Bamberger, Zdeněk Kasner, Alice Rueda, Amanda Pestana, Amir Feizpour, Ammar Khan, Amy Faranak, Ana Santos, Anthony Hevia, Antigona Unldreaj, Arash Aghagol, Arezoo Abdollahi, Aycha Tammour, Azadeh HajiHosseini, Bahareh Behroozi, Benjamin Ajibade, Bharat Saxena, Carlos Muñoz Ferrandis, Danish Contractor, David Lansky, Davis David, Douwe Kiela, Duong A. Nguyen, Edward Tan, Emi Baylor, Ezinwanne Ozoani, Fatima Mirza, Frankline Ononiwu, Habib Rezanejad, Hessie Jones, Indrani Bhattacharya, Irene Solaiman, Irina Sedenko, Isar Nejadgholi, Jesse Passmore, Josh Seltzer, Julio Bonis Sanz, Livia Dutra, Mairon Samagaio, Maraim Elbadri, Margot Mieskes, Marissa Gerchick, Martha Akinlolu, Michael McKenna, Mike Qiu, Muhammed Ghauri, Mykola Burynok, Nafis Abrar, Nazneen Rajani, Nour Elkott, Nour Fahmy, Olanrewaju Samuel, Ran An, Rasmus Kromann, Ryan Hao, Samira Alizadeh, Sarmad Shubber, Silas Wang, Sourav Roy, Sylvain Viguier, Thanh Le, Tobi Oyebade, Trieu Le, Yoyo Yang, Zach Nguyen, Abhinav Ramesh Kashyap, Alfredo Palasciano, Alison Callahan, Anima Shukla, Antonio Miranda-Escalada, Ayush Singh, Benjamin Beilharz, Bo Wang, Caio Brito, Chenxi Zhou, Chirag Jain, Chuxin Xu, Clémentine Fourrier, Daniel León Periñán, Daniel Molano, Dian Yu, Enrique Manjavacas, Fabio Barth, Florian Fuhrimann, Gabriel Altay, Giyaseddin Bayrak, Gully Burns, Helena U. Vrabec, Imane Bello, Ishani Dash, Jihyun Kang, John Giorgi, Jonas Golde, Jose David Posada, Karthik Rangasai Sivaraman, Lokesh Bulchandani, Lu Liu, Luisa Shinzato, Madeleine Hahn de Bykhovetz, Maiko Takeuchi, Marc Pàmies, Maria A Castillo, Marianna Nezhurina, Mario Sänger, Matthias Samwald, Michael Cullan, Michael Weinberg, Michiel De Wolf, Mina Mihaljcic, Minna Liu, Moritz Freidank, Myungsun Kang, Natasha Seelam, Nathan Dahlberg, Nicholas Michio Broad, Nikolaus Muellner, Pascale Fung, Patrick Haller, Ramya Chandrasekhar, Renata Eisenberg, Robert Martin, Rodrigo Canalli, Rosaline Su, Ruisi Su, Samuel Cahyawijaya, Samuele Garda, Shlok S Deshmukh, Shubhanshu Mishra, Sid Kiblawi, Simon Ott, Sinee Sang-aroonsiri, Srishti Kumar, Stefan Schweter, Sushil Bharati, Tanmay Laud, Théo Gigant, Tomoya Kainuma, Wojciech Kusa, Yanis Labrak, Yash Shailesh Bajaj, Yash Venkatraman, Yifan Xu, Yingxin Xu, Yu Xu, Zhe Tan, Zhongli Xie, Zifan Ye, Mathilde Bras, Younes Belkada, and Thomas Wolf. 2022. BLOOM: a 176b-parameter open-access multilingual language model.

Maxime Bouthors, Josep Crego, and François Yvon. 2023. Towards example-based NMT with multi-Levenshtein transformers. In *Proceedings of the 2023 Conference on Empirical Methods in Natural Language Processing*, pages 1830–1846, Singapore. Association for Computational Linguistics.

Lynne Bowker. 2002. *Computer-aided translation technology: A practical introduction*. University of Ottawa Press.

Bram Bulte and Arda Tezcan. 2019. Neural fuzzy repair: Integrating fuzzy matches into neural machine translation. In *Proceedings of the 57th Annual Meeting of the Association for Computational Linguistics*, pages 1800–1809, Florence, Italy. Association for Computational Linguistics.

Deng Cai, Yan Wang, Huayang Li, Wai Lam, and Lemao Liu. 2021. Neural machine translation with monolingual translation memory. In *Proceedings of the 59th Annual Meeting of the Association for Computational Linguistics and the 11th International Joint Conference on Natural Language Processing (Volume 1: Long Papers)*, pages 7307–7318, Online. Association for Computational Linguistics.

Michael Carl, Andy Way, and Walter Daelemans. 2004. Recent advances in example-based machine translation. *Computational Linguistics*, 30:516–520.

Xin Cheng, Shen Gao, Lemao Liu, Dongyan Zhao, and Rui Yan. 2022. Neural machine translation with contrastive translation memories. In *Proceedings of the 2022 Conference on Empirical Methods in Natural Language Processing*, pages 3591–3601, Abu Dhabi, United Arab Emirates. Association for Computational Linguistics.

M. Amin Farajian, Marco Turchi, Matteo Negri, and Marcello Federico. 2017. Multi-domain neural machine translation through unsupervised adaptation. In *Proceedings of the Second Conference on Machine Translation*, pages 127–137, Copenhagen, Denmark. Association for Computational Linguistics.

Jade Goldstein and Jaime Carbonell. 1998. Summarization: (1) using MMR for diversity- based reranking and (2) evaluating summaries. In *TIPSTER TEXT PROGRAM PHASE III: Proceedings of a Workshop held at Baltimore, Maryland, October 13-15, 1998*, pages 181–195, Baltimore, Maryland, USA. Association for Computational Linguistics.

Jiatao Gu, Changhan Wang, and Junbo Zhao. 2019. Levenshtein transformer. In *Advances in Neural Information Processing Systems*, volume 32. Curran Associates, Inc.

Jiatao Gu, Yong Wang, Kyunghyun Cho, and Victor O.K. Li. 2018. Search Engine Guided Neural Machine Translation. *Proceedings of the AAAI Conference on Artificial Intelligence*, 32(1).

Shivanshu Gupta, Matt Gardner, and Sameer Singh. 2023. Coverage-based example selection for in-context learning. In *Findings of the Association for Computational Linguistics: EMNLP 2023*, pages 13924–13950, Singapore. Association for Computational Linguistics.

Junxian He, Graham Neubig, and Taylor Berg-Kirkpatrick. 2021a. Efficient nearest neighbor language models. In *Proceedings of the 2021 Conference on Empirical Methods in Natural Language Processing*, pages 5703–5714, Online and Punta Cana, Dominican Republic. Association for Computational Linguistics.

Qiuxiang He, Guoping Huang, Qu Cui, Li Li, and Lemao Liu. 2021b. Fast and accurate neural machine translation with translation memory. In *Proceedings of the 59th Annual Meeting of the Association for Computational Linguistics and the 11th International Joint Conference on Natural Language Processing (Volume 1: Long Papers)*, pages 3170–3180, Online. Association for Computational Linguistics.

Amr Hendy, Mohamed Abdelrehim, Amr Sharaf, Vikas Raunak, Mohamed Gabr, Hitokazu Matsushita, Young Jin Kim, Mohamed Afify, and Hany Hassan Awadalla. 2023. How good are GPT models at machine translation? a comprehensive evaluation. *CoRR*, abs/2302.09210.

Edward J Hu, yelong shen, Phillip Wallis, Zeyuan Allen-Zhu, Yuanzhi Li, Shean Wang, Lu Wang, and Weizhu Chen. 2022. LoRA: Low-rank adaptation of large language models. In *International Conference on Learning Representations*.

Urvashi Khandelwal, Angela Fan, Dan Jurafsky, Luke Zettlemoyer, and Mike Lewis. 2021. Nearest Neighbor Machine Translation. In *Proceedings of the International Conference on Learning Representations*.

Philipp Koehn and Rebecca Knowles. 2017. Six challenges for neural machine translation. In *Proceedings of the First Workshop on Neural Machine Translation*, pages 28–39, Vancouver. Association for Computational Linguistics.

Philipp Koehn and Jean Senellart. 2010. Convergence of translation memory and statistical machine translation. In *Proceedings of the Second Joint EM+/CNGL Workshop: Bringing MT to the User: Research on Integrating MT in the Translation Industry*, pages 21–32, Denver, Colorado, USA. Association for Machine Translation in the Americas.

Hugo Laurençon, Lucile Saulnier, Thomas Wang, Christopher Akiki, Albert Villanova del Moral, Teven Le Scao, Leandro Von Werra, Chenghao Mou, Eduardo González Ponferrada, Huu Nguyen, et al. 2022. The BigScience ROOTS Corpus: A 1.6 TB Composite Multilingual Dataset. In *Thirty-sixth Conference on Neural Information Processing Systems Datasets and Benchmarks Track*.

Huayang Li, Yixuan Su, Deng Cai, Yan Wang, and Lemao Liu. 2022a. A survey on retrieval-augmented text generation. *CoRR*, abs/2202.01110.

Huayang Li, Yixuan Su, Deng Cai, Yan Wang, and Lemao Liu. 2022b. A survey on retrieval-augmented text generation.

Aswanth M, Ratish Puduppully, Raj Dabre, and Anoop Kunchukuttan. 2023. CTQScorer: Combining multiple features for in-context example selection for machine translation. In *Findings of the Association for Computational Linguistics: EMNLP 2023*, pages 7736–7752, Singapore. Association for Computational Linguistics.

Pedro Henrique Martins, Zita Marinho, and André F. T. Martins. 2022. Chunk-based nearest neighbor machine translation. In *Proceedings of the 2022 Conference on Empirical Methods in Natural Language Processing*, pages 4228–4245, Abu Dhabi, United Arab Emirates. Association for Computational Linguistics.

Yuxian Meng, Xiaoya Li, Xiayu Zheng, Fei Wu, Xiaofei Sun, Tianwei Zhang, and Jiwei Li. 2022. Fast nearest neighbor machine translation. In *Findings of the Association for Computational Linguistics: ACL 2022*, pages 555–565, Dublin, Ireland. Association for Computational Linguistics.

Yasmin Moslem, Rejwanul Haque, John D. Kelleher, and Andy Way. 2023. Adaptive machine translation with large language models. In *Proceedings of the 24th Annual Conference of the European Association for Machine Translation*, pages 227–237, Tampere, Finland. European Association for Machine Translation.

Yongyu Mu, Abudurexiti Reheman, Zhiquan Cao, Yuchun Fan, Bei Li, Yinqiao Li, Tong Xiao, Chunliang Zhang, and Jingbo Zhu. 2023. Augmenting large language model translators via translation memories. In *Findings of the Association for Computational Linguistics: ACL 2023*, pages 10287–10299, Toronto, Canada. Association for Computational Linguistics.

Makoto Nagao. 1984. A framework of a mechanical translation between Japanese and English by analogy principle. In *Artificial and human intelligence*. Elsevier Science Publishers. B.V.

Ayana Niwa, Sho Takase, and Naoaki Okazaki. 2022. Nearest neighbor non-autoregressive text generation. *CoRR*, abs/2208.12496.

Kishore Papineni, Salim Roukos, Todd Ward, and Wei-Jing Zhu. 2002. Bleu: a method for automatic evaluation of machine translation. In *Proceedings of the 40th Annual Meeting of the Association for Computational Linguistics*, pages 311–318, Philadelphia, Pennsylvania, USA. Association for Computational Linguistics.

Minh Quang Pham, Jitao Xu, Josep Crego, François Yvon, and Jean Senellart. 2020. Priming neural machine translation. In *Proceedings of the Fifth Conference on Machine Translation*, pages 516–527, Online. Association for Computational Linguistics.

Matt Post. 2018. A call for clarity in reporting BLEU scores. In *Proceedings of the Third Conference on Machine Translation: Research Papers*, pages 186–191, Brussels, Belgium. Association for Computational Linguistics.

Abudurexiti Reheman, Tao Zhou, Yingfeng Luo, Di Yang, Tong Xiao, and Jingbo Zhu. 2023. Prompting neural machine translation with translation memories. *Proceedings of the AAAI Conference on Artificial Intelligence*, 37(11):13519–13527.

Ricardo Rei, José G. C. de Souza, Duarte Alves, Chrysoula Zerva, Ana C Farinha, Taisiya Glushkova, Alon Lavie, Luisa Coheur, and André F. T. Martins. 2022. COMET-22: Unbabel-IST 2022 submission for the metrics shared task. In *Proceedings of the Seventh Conference on Machine Translation (WMT)*, pages 578–585, Abu Dhabi, United Arab Emirates (Hybrid). Association for Computational Linguistics.

Stephen Robertson and Hugo Zaragoza. 2009. The probabilistic relevance framework: BM25 and beyond. *Foundations and Trends® in Information Retrieval*, 3(4):333–389.

Cynthia Rudin. 2019. Stop explaining black box machine learning models for high stakes decisions and use interpretable models instead. *Nature Machine Intelligence*, 1(5):206–215.

Suzanna Sia and Kevin Duh. 2023a. In-context learning as maintaining coherency: A study of on-the-fly machine translation using large language models. In *Proceedings of Machine Translation Summit XIX, Vol. 1: Research Track*, pages 173–185, Macau SAR, China. Asia-Pacific Association for Machine Translation.

Suzanna Sia and Kevin Duh. 2023b. In-context learning as maintaining coherency: A study of on-the-fly machine translation using large language models.

Harold Somers. 1999. Review article: Example-based machine translation. *Machine Translation*, 14(2):113–157.

Jörg Tiedemann. 2012. Parallel data, tools and interfaces in OPUS. In *Proceedings of the Eighth International Conference on Language Resources and Evaluation (LREC'12)*, pages 2214–2218, Istanbul, Turkey. European Language Resources Association (ELRA).

Hugo Touvron, Thibaut Lavril, Gautier Izacard, Xavier Martinet, Marie-Anne Lachaux, Timothée Lacroix, Baptiste Rozière, Naman Goyal, Eric Hambro, Faisal Azhar, Aurelien Rodriguez, Armand Joulin, Edouard Grave, and Guillaume Lample. 2023. Llama: Open and efficient foundation language models.

David Vilar, Markus Freitag, Colin Cherry, Jiaming Luo, Viresh Ratnakar, and George Foster. 2023. Prompting PaLM for translation: Assessing strategies and performance. In *Proceedings of the 61st Annual Meeting of the Association for Computational Linguistics (Volume 1: Long Papers)*, pages 15406–15427, Toronto, Canada. Association for Computational Linguistics.

Mengzhou Xia, Guoping Huang, Lemao Liu, and Shuming Shi. 2019. Graph based translation memory for neural machine translation. *Proceedings of the AAAI Conference on Artificial Intelligence*, 33(01):7297–7304.

Jitao Xu, Josep Crego, and Jean Senellart. 2020. Boosting neural machine translation with similar translations. In *Proceedings of the 58th Annual Meeting of the Association for Computational Linguistics*, pages 1580–1590, Online. Association for Computational Linguistics.

Jitao Xu, Josep Crego, and Jean Senellart. 2022. Boosting neural machine translation with similar translations. In *Proceedings of the 15th Biennial Conference of the Association for Machine Translation in the Americas (Volume 2: Users and Providers Track and Government Track)*, pages 282–292, Orlando, USA. Association for Machine Translation in the Americas.

Jitao Xu, Josep Crego, and François Yvon. 2023. Integrating translation memories into non-autoregressive machine translation. In *Proceedings of the 17th Conference of the European Chapter of the Association for Computational Linguistics*, pages 1326–1338, Dubrovnik, Croatia. Association for Computational Linguistics.

Biao Zhang, Barry Haddow, and Alexandra Birch. 2023. Prompting large language model for machine translation: A case study. In *Proceedings of the 40th International Conference on Machine Learning*, ICML'23. JMLR.org.

Jingyi Zhang, Masao Utiyama, Eiichro Sumita, Graham Neubig, and Satoshi Nakamura. 2018. Guiding neural machine translation with retrieved translation pieces. In *Proceedings of the 2018 Conference of the North American Chapter of the Association for Computational Linguistics: Human Language Technologies, Volume 1 (Long Papers)*, pages 1325–1335, New Orleans, Louisiana. Association for Computational Linguistics.

Kangjie Zheng, Longyue Wang, Zhihao Wang, Binqi Chen, Ming Zhang, and Zhaopeng Tu. 2023. Towards a unified training for Levenshtein transformer. In *Proceedings of the IEEE International Conference on Acoustics, Speech and Signal Processing (ICASSP)*, pages 1–5.

Xin Zheng, Zhirui Zhang, Junliang Guo, Shujian Huang, Boxing Chen, Weihua Luo, and Jiajun Chen. 2021. Adaptive nearest neighbor machine translation. In *Proceedings of the 59th Annual Meeting of the Association for Computational Linguistics and the 11th International Joint Conference on Natural Language Processing (Volume 2: Short Papers)*, pages 368–374, Online. Association for Computational Linguistics.

A δ-LCS formulation

The general formulation of a similarity metric based on the edit distance is:

$$\mathrm{sim}(\tilde{\mathbf{x}}, \mathbf{x}) = 1 - \frac{\Delta(\tilde{\mathbf{x}}, \mathbf{x})}{N(|\tilde{\mathbf{x}}|, |\mathbf{x}|)}, \tag{4}$$

with N a normalizing term that upper bounds Δ. Δ is parameterized by d, a, r, corresponding respectively to the delete (resp. insert, replace) costs (assuming copies have a 0 cost), generalizing the standard ED setting ($d = a = r = 1$). Δ corresponds to the minimal generalized cost necessary to edit $\tilde{\mathbf{x}}$ into $\mathbf{x}$.

We define an Edit Common Subsequence $\mathrm{ecs}_{d,a,r}(\tilde{\mathbf{x}}, \mathbf{x})$, which corresponds to the tokens copied when optimally editing $\tilde{\mathbf{x}}$ into $\mathbf{x}$.[16] If $a + d \leq r$, then the copied tokens are a Longest Common Subsequence (LCS) of $\tilde{\mathbf{x}}$ and $\mathbf{x}$: $\mathrm{ecs}_{d,a,r}(\tilde{\mathbf{x}}, \mathbf{x}) = \mathrm{lcs}(\tilde{\mathbf{x}}, \mathbf{x})$.

If $a + d > r$:

$$\Delta(\tilde{\mathbf{x}}, \mathbf{x}) = \begin{cases} r\,[|\mathbf{x}| - |ecs_{d,a,r}(\tilde{\mathbf{x}}, \mathbf{x})|] + d(|\tilde{\mathbf{x}}| - |\mathbf{x}|) \\ \quad \text{if } |\mathbf{x}| \leq |\tilde{\mathbf{x}}| \\ r\,[|\tilde{\mathbf{x}}| - |ecs_{d,a,r}(\tilde{\mathbf{x}}, \mathbf{x})|] + a(|\mathbf{x}| - |\tilde{\mathbf{x}}|) \\ \quad \text{if } |\mathbf{x}| > |\tilde{\mathbf{x}}| \end{cases} \tag{5}$$

If $a + d \leq r$:

$$\Delta(\tilde{\mathbf{x}}, \mathbf{x}) = a(|\mathbf{x}| - |\,\mathrm{lcs}(\tilde{\mathbf{x}}, \mathbf{x})|) + d(|\tilde{\mathbf{x}}| - |\,\mathrm{lcs}(\tilde{\mathbf{x}}, \mathbf{x})|) \tag{6}$$

Δ is maximal when $|\,\mathrm{ecs}\,| = 0$. The normalization term can be expressed as:

$$N_{d,a,r}(|\tilde{\mathbf{x}}|, |\mathbf{x}|) = \begin{cases} a|\mathbf{x}| + d|\tilde{\mathbf{x}}| \\ \quad \text{if } a + d \leq r \\ (r-d)|\mathbf{x}| + d|\tilde{\mathbf{x}}| \\ \quad \text{if } a + d > r \text{ and } |\mathbf{x}| \leq |\tilde{\mathbf{x}}| \\ (r-a)|\tilde{\mathbf{x}}| + a|\mathbf{x}| \\ \quad \text{if } a + d > r \text{ and } |\mathbf{x}| > |\tilde{\mathbf{x}}| \end{cases} \tag{7}$$

Choosing $(d, a, r) = (0, 1, 1)$ makes $\mathrm{sim}(\tilde{\mathbf{x}}, \mathbf{x})$ compute the coverage of $\mathbf{x}$ with tokens from $\tilde{\mathbf{x}}$:

$$\mathrm{sim}(\tilde{\mathbf{x}}, \mathbf{x}) = 1 - \frac{a(|\mathbf{x}| - |\,\mathrm{lcs}(\tilde{\mathbf{x}}, \mathbf{x})|)}{a|\mathbf{x}|} = \frac{|\,\mathrm{lcs}(\tilde{\mathbf{x}}, \mathbf{x})|}{|\mathbf{x}|} \tag{8}$$

B Fuzzy-Matching detailed settings

Note that the code used for our experiments is open source.[17] Additional details can be found in the repository.

B.1 NGM

For a source q, n-gram matching identifies $g(q, x)$, the longest common n-gram between q and any source example x. x passes the filter if (1) $|g(q, x)| \geq$ ML, an absolute length threshold that we choose to be 3; (2) $|g(q, x)| \geq \tau|q|$, with τ varying between 0.3 (following the works of Xu et al. (2022); Bouthors et al. (2023)) and 0.2 – which is a more permissive filter that increases the chance to having higher scoring matches w.r.t. the ranker (BM25, ED).

Our algorithm uses a suffix array structure which directly indexes the n-grams and enables to search for n-gram matches in the sorted array with a logarithmic complexity.

However, the distribution of the number of candidates passing the filter has a high variance (see Table 9). In consequence, it can be ineffective to select only a small amount of relevant candidates

[16]There can be in fact several concurrent common subsequences.

[17]Available at https://github.com/SYSTRAN/fuzzy-match

(too permissive). On the other hand, it can often retrieve very few candidates (<3).

τ	Q1	Q2	Q3
0.2	6	116	1573
0.3	2	15	293
0.4	1	6	99
0.5	1	4	54

Table 9: Average 1st, 2nd, 3rd quartiles for the eleven datasets of the distribution of candidates passing the NGM filter w.r.t. τ.

B.2 BM25

BM25 (Robertson and Zaragoza, 2009) is a widely used ranker in Information Retrieval. It often competes with state-of-the-art neural retrievers and is used as a baseline. It is a fully unsupervised method based on computations inspired by the *tf-idf* scoring function. BM25 typically aims to retrieve documents that have common rare words with the query.

Our implementation is not as optimized as in Elasticsearch[18] or Apache Lucene[19], it is nonetheless very efficient and integrates seamlessly in our translation pipeline.

Since computing BM25 is computationally linear in the number of sentences in the TM, naive implementations can sometimes be slow. We use an *inverted index*, which directly maps each term to the set of TM segments where they occur. Instead of computing BM25 for every sentence, we only do it for the union of TM segments containing query terms. As common terms such as punctuation, prepositions, etc. will likely map to most TMs, they are removed from the index.

In our implementation, a term is said to be common if it appears in more than $p\%$ of the segments. We find that p can be quite small (2%) without affecting the set of retrieved sentences too much. For instance, we find that BM25+LCS with the setting $L = 100, p = 2\%$ has 90.4% Jaccard similarity (in terms of the set of indices of retrieved segments) with the setting $L = 100, p = 10\%$.

In preliminary experiments, we also varied the value of L (10 vs 100) and found out that a low L negatively affects the model-agnostic metrics (coverage, relevance) and set $L = 100$ in all experiments.

Table 10 contains the model-agnostic scores on the training set en-fr for filtering pipelines. It suggests that BM25 filter is better at increasing coverage and relevance than NGM. But it has a higher latency. This can explain why BM25-filtered pipelines are better at training time (as shown in Appendix D).

[18]https://elastic.co.

[19]http:lucene.apache.org.

C Models detailed settings

C.1 NFA

We made our own implementation of NFA based on the work of Bulte and Tezcan (2019) with the following parameters: size of word embedding: 512; size of hidden layers: 512; size of inner feed forward layer: $2,048$; number of heads: 8; number of layers: 6; batch size: $4,096$ tokens. We set warmup steps to $4,000$ and update learning rate for every 8 iterations. Fine-tuning is performed continuing Adam with learning rate decay schedule until convergence. The models are trained with one NVIDIA P100 GPU. We use a joint vocabulary of 32K for both source and target sides. At inference we use a beam size of 5.

C.2 `TM`^{N}`-LevT`

We use a Multi-Levenshtein Transformer architecture with embeddings of dimension 512; feedforward layers of size 2048; number of heads 8; number of encoder and decoder layers: 6; shared-embeddings; dropout: 0.3.

During training, we use Adam optimizer with (β_1, β_2)=(0.9, 0.98); inverse sqrt scheduler; learning rate: $5e^{-4}$; label smoothing: 0.1; warmup updates: 10,000; float precision: 16. We fixed the number of iterations at 60k. The batch size and number of GPUs are set to have, on average, ~ 450 samples per iteration. We performed a pretraining of the model on a synthetic dataset as described by Bouthors et al. (2023). We use a joint vocabulary of 32K. For decoding, we use realignment and iterative refinement with an empty placeholder penalty of 3, and a max number of iterations of 10 (Gu et al., 2019).

C.3 BLOOM

BLOOM is a family of large open-source causal multilingual language models trained in the course of the BigScience project (BigScience et al., 2022). Our experiments use the largest available version, comprising 176b parameters. This model has been repeatedly evaluated in translation scenarios see e.g. (Bawden and Yvon, 2023). BLOOM's training corpus officially contains a fair share of English and French, but hardly any German[20] (Laurençon et al., 2022). We thus consider the two following translation directions: en-fr and de-en, selecting target languages for which the generation abilities are strong.

Following common practice, we select a simple prompt of the form "[src]: source sentence. =[trg]: target sentence", where "[src]" and "[trg]" are language tags denoting respectively the source and the target sentences, using either 1 or 3 in-context examples before the source query. All experiments generate translation in pure greedy mode; generation stops either when the end of sentence token is produced or when a maximum length of 256 tokens is reached. As is also custom for BLOOM, which tends to overgenerate, we post-process the output to shorten excessively long outputs and make sure the target is never longer than 1.5 times the source (measured in chars). This post-processing only impacts about 5 to 10% of all output sentences.

D Cross inference for `TM`^{3}`-LevT`

We train Multi-Levenshtein Transformer (`TM`^{3}`-LevT`) with a series of retrieval settings with NGM (Table 11) and BM25 filters (Table 12). Afterward, each trained model is evaluated on all the retrieval settings of the same filter category. This way, it is possible to identify which train/inference setting pairs perform better and whether having the same setting for training and inference is recommended. We can, for both NGM and BM25, identify a systematically independent optimal training and inference setting. We find that the BM25 filter is slightly better in general. The contrastive method only helps at inference time and for medium matches, suggesting adapting the factor to the fuzzy score (ED).

0.1-LCS is overall better, except for training with BM25, which surprisingly achieves the best scores with LED. Probably, having more source-similar sentences at train time is necessary to compensate for the unordered nature of BM25, required by `TM`^{3}`-LevT`.

E Per domain analysis

Tables 13,14,15 contain the detailed per-domain analysis for direction en-fr. We insist on the fact that this corpus offers a high variability across domains (size, density, sentence length...). Low-density domains (Ubuntu, News-Commentary) and high-density ones (EMEA, KDE4) are the ones the most likely to differ from the average best. Moreover, low-density domains seem to prefer diversity and coverage with BLOOM. As for `TM`^{3}`-LevT`, the lowest and highest densities are both more inclined to choose pipelines with filters. The model may suffer from low-quality examples from a difficult low-density domain while not necessitating diversity in the case of dense ones.

F Illustration

Figure 2 illustrates the variability across retrieval settings. We can observe their main characteristics:

- Since NGM and BM25 act here as filters, they barely change the retrieved set.
- δ-LCS covers more words in the source at the expense of longer sentences.
- The first retrieved sentence for the contrastive ranking is always the same as for LED, but the other two are often more diverse, covering more terms.

[20]Even though traces of German can be found in the corpus.

Ngram filter ($\tau =$)	0.3	0.2	0.2	0.3	0.2	-	-	-	-
BM25 filter ($L =$ or '-')	-	-	-	-	-	100	100	100	100
Ranking (LED, LCS, δ-LCS)	LED	LCS	δ-LCS	δ-LCS	δ-LCS	LED	LCS	δ-LCS	δ-LCS
Contrast factor ($\alpha =$)	-	-	-	-	0.3	-	-	-	0.3
coverage	27.9	42.5	38.8	35.0	39.5	32.6	**48.1**	43.6	44.6
relevance	23.0	23.3	26.1	23.9	24.6	26.6	28.8	**29.2**	27.6
length	15.4	40.6	24.0	22.0	25.5	**15.2**	30.1	22.4	23.5

Table 10: Retrieval scores averaged over 11 domains (train sets, en-fr).

source		Most patients required treatment for their orthostatic hypotension.
NGM+LED		*coverage:* 3 words
	1	Rare: peripheral coldness, **orthostatic hypotension.**
	2	Entacapone may aggravate levodopa-induced **orthostatic hypotension.**
	3	- Stalevo may induce **orthostatic hypotension.**
BM25+LED		*coverage:* 3 words
	1	Hypotension, **orthostatic hypotension.**
	2	- Stalevo may induce **orthostatic hypotension.**
	3	Rare: peripheral coldness, **orthostatic hypotension.**
LED		*coverage:* 3 words
	1	Hypotension, **orthostatic hypotension.**
	2	- Stalevo may induce **orthostatic hypotension.**
	3	Rare: peripheral coldness, **orthostatic hypotension.**
δ-LCS		*coverage:* 6 words
	1	Patients receiving aripiprazole solution **for** injection should be observed for **orthostatic hypotension.**
	2	If parenteral benzodiazepine therapy is deemed necessary in addition to aripiprazole solution for injection, **patients** should be monitored **for** excessive sedation and for **orthostatic hypotension** (see section 4.5)**.**
	3	Hypotension VELCADE **treatment** is commonly associated with **orthostatic/** postural **hypotension.**
LED$_c$		*coverage:* 6 words
	1	Hypotension, **orthostatic hypotension.**
	2	**Most patients** had relief of symptoms after stopping **treatment.**
	3	Entacapone may aggravate levodopa-induced **orthostatic hypotension.**
δ-LCS$_c$		*coverage:* 5 words
	1	Patients receiving aripiprazole solution **for** injection should be observed for **orthostatic hypotension.**
	2	A minority of **patients** with **orthostatic hypotension** experienced syncopal events**.**
	3	If parenteral benzodiazepine therapy is deemed necessary in addition to aripiprazole solution **for** injection, **patients** should be monitored **for** excessive sedation and for **orthostatic hypotension** (see section 4.5)**.**

Figure 2: Illustration of the variability across some retrieval settings and their respective coverage for a source sentence from EMEA. For each setting, we represent the source-side 3 best-ranked sentences.

train \ infer	LED$^+$	LCS$^-$	δ-LCS$^-$	δ-LCS$^+$	δ-LCS$_c^-$	LED$^+$	LCS$^-$	δ-LCS$^-$	δ-LCS$^+$	δ-LCS$_c^-$
LED$^+$	43.9	44.1	**44.8**	**44.8**	**44.8**	56.0	55.6	56.2	**56.3**	56.1
LCS$^-$	43.9	44.5	44.8	44.6	**45.0**	55.7	56.2	56.5	56.3	**56.6**
δ-LCS$^-$	44.6	44.8	**45.3**	45.2	**45.5**	55.9	56.1	**56.6**	56.5	56.5
δ-LCS$^+$	44.1	44.5	45.3	45.1	**45.3**	56.3	56.3	**56.9**	**56.9**	56.8
δ-LCS$_c^-$	43.8	44.1	44.7	44.5	**44.9**	55.6	55.9	**56.4**	56.2	56.3

Table 11: Average cross-inference BLEU score accross all domains for *test-0.4* (left) and *test-0.6* (right) with NGM filter. τ is specified with $^+$ for 0.3 and $^-$ for 0.2; c denotes contrast. Best column-wise (resp. row-wise) BLEU are underlined (resp. **in bold**).

train \ infer	LED	LCS	δ-LCS	δ-LCS$_c$	LED	LCS	δ-LCS	δ-LCS$_c$
LED	45.1	45.2	**45.7**	**45.7**	57.0	56.5	**57.1**	57.0
LCS	44.7	45.1	45.3	**45.4**	56.1	56.4	**56.5**	56.4
δ-LCS	44.6	45.1	45.3	**45.4**	56.4	56.6	56.8	**56.9**
δ-LCS$_c$	42.3	43.2	43.6	**44.0**	55.2	55.6	55.8	**56.1**

Table 12: Average cross-inference BLEU score accross all domains for *test-0.4* (left) and *test-0.6* (right) with BM25 filter. c designates contrast. Best column-wise (resp. row-wise) BLEU are underlined (resp. **in bold**).

pipeline	ECB	EME	Epp	GNO	JRC	KDE	News	PHP	TED	Ubu	Wiki	**all**
test-0.4												
NGM+LED 1-1	64.7	65.3	44.9	**71.2**	76.4	**65.9**	**30.4**	42.0	43.9	57.5	43.5	**55.1**
NGM+LED 3-3	64.8	65.8	44.6	71.1	76.4	65.4	29.7	41.4	**44.0**	56.4	44.4	54.9
NGM+LED 3-2	64.9	65.6	44.5	70.8	76.3	65.4	29.7	41.4	**44.0**	56.3	44.2	54.8
NGM+LED 3-1	64.4	65.0	44.7	70.6	76.2	65.7	29.8	41.6	43.9	56.6	44.1	54.8
BM25	64.1	66.3	44.8	70.8	76.0	65.0	30.2	40.0	43.5	**57.7**	43.0	54.7
BM25$_c$	64.2	**66.4**	44.8	70.9	76.0	64.9	30.2	40.0	43.5	57.6	43.0	54.7
LED	64.8	65.4	44.8	71.1	**76.5**	65.4	29.6	41.0	43.8	56.7	**44.5**	54.9
δ-LCS	65.0	64.6	44.8	70.9	**76.5**	65.3	29.9	41.4	43.6	57.3	43.9	54.8
δ-LCS$_c$	**65.2**	63.9	**45.0**	70.8	76.4	65.4	29.8	**41.8**	43.9	57.3	43.6	54.8
test-0.6												
NGM+LED 1-1	71.0	**73.3**	59.1	79.9	83.7	68.1	**27.6**	**46.4**	63.7	64.4	69.8	64.3
NGM+LED 3-3	70.8	73.1	59.1	80.4	**83.9**	**69.4**	27.3	45.7	64.2	64.1	69.7	64.3
NGM+LED 3-2	70.8	72.3	59.1	80.4	83.7	69.2	27.2	45.6	64.1	63.8	**70.1**	64.2
NGM+LED 3-1	70.1	72.2	59.0	79.8	83.6	68.5	27.4	44.8	64.1	63.5	69.5	63.9
BM25	**71.2**	72.8	**59.2**	81.0	83.5	67.7	27.5	44.8	**64.3**	**64.9**	69.1	64.2
BM25$_c$	**71.2**	72.9	**59.2**	**81.1**	83.5	67.7	**27.6**	44.8	**64.3**	64.8	68.9	64.2
LED	70.8	73.1	59.1	80.4	83.8	**69.4**	27.3	45.7	64.1	64.2	69.9	**64.4**
δ-LCS	71.1	73.2	59.1	80.1	83.7	68.7	27.2	46.0	64.2	64.4	69.7	64.3
δ-LCS$_c$	71.1	73.0	59.1	80.2	83.8	68.9	27.1	45.9	**64.3**	64.4	69.5	64.3

Table 13: BLEU score (en-fr): NFA trained on NGM+LED ($\tau = 0.3$), inferred on filter-free retrieval settings.

pipeline	ECB	EME	Epp	GNO	JRC	KDE	News	PHP	TED	Ubu	Wiki	**all**
test-0.4												
best NGM+ED pair	57.4	**57.6**	35.8	62.0	68.7	55.9	**22.1**	29.2	31.6	45.2	**35.8**	45.6
best BM25+ED pair	57.5	57.0	35.8	63.0	69.2	**57.5**	21.9	30.4	31.3	45.5	34.8	45.8
LED	56.7	56.9	35.6	62.4	68.8	56.4	21.5	30.6	32.3	45.9	34.4	45.6
δ-LCS	57.3	56.5	**36.8**	**63.3**	69.2	56.7	21.9	**30.9**	**31.9**	45.1	34.9	45.9
δ-LCS$_c$	**57.7**	56.8	36.4	62.8	**69.3**	56.8	21.8	30.8	31.7	**46.5**	35.2	**46.0**
test-0.6												
best NGM+ED pair	65.8	**68.3**	51.6	73.3	77.5	**63.0**	21.2	33.4	55.9	53.0	63.6	57.0
best BM25+ED pair	66.7	67.8	51.2	72.6	78.4	62.5	**21.4**	34.9	55.9	53.7	63.6	57.2
LED	66.2	68.1	51.5	73.0	78.4	62.7	**21.4**	34.7	55.5	**54.0**	**64.3**	57.3
δ-LCS	**66.8**	**68.3**	51.6	**73.3**	78.4	62.9	21.3	**35.4**	**56.2**	53.5	63.9	**57.4**
δ-LCS$_c$	66.6	68.1	**52.0**	72.5	**78.6**	62.0	21.3	35.0	55.3	53.9	63.5	57.2

Table 14: BLEU score (en-fr): TM^3-LevT with the best train/infer pipelines (top) or trained on BM25+LED and inferred on filter-free ED variants.

model	k-shot	ECB	EME	GNO	KDE	News	PHP	Ubu	**all**
test-0.4									
random	1	26.2	27.8	37.0	27.4	24.8	19.3	42.4	29.3
random	3	30.7	34.7	40.1	31.8	26.9	23.4	44.2	33.1
LED	1	50.7	54.4	57.6	48.0	24.2	31.3	54.5	45.8
LED	3	52.7	56.7	59.0	50.2	26.1	33.0	57.2	47.8
LED$_c$	3	**53.6**	**57.6**	**59.9**	**51.3**	26.3	33.0	56.7	**48.3**
BM25	1	48.2	57.2	57.7	47.0	25.2	31.1	53.9	45.8
BM25	3	52.2	58.9	59.1	49.7	27.0	**33.2**	56.7	48.1
BM25$_c$	3	52.7	58.7	59.4	49.5	**27.3**	32.4	56.6	48.1
δ-LCS	3	52.8	56.6	59.0	50.9	26.4	33.1	57.8	48.1
δ-LCS$_c$	3	52.3	57.4	58.8	51.0	26.9	33.1	**58.1**	48.2
test-0.6									
random	1	26.7	26.6	33.9	25.3	21.9	18.7	43.9	28.1
random	3	32.1	34.1	36.5	29.3	24.3	23.2	47.5	32.4
LED	1	57.9	64.7	58.7	52.9	21.5	32.8	61.4	50.0
LED	3	60.7	64.6	60.3	53.8	23.4	34.8	63.2	51.5
LED$_c$	3	**61.5**	65.1	**61.0**	54.1	23.7	**35.2**	63.5	**52.0**
BM25	1	57.5	63.8	58.7	50.5	22.0	32.4	62.0	49.6
BM25	3	59.7	65.2	60.8	53.9	24.3	35.0	64.0	51.8
BM25$_c$	3	59.6	64.9	60.3	53.7	**24.1**	34.1	63.0	51.4
δ-LCS	3	59.6	**65.3**	60.8	54.5	23.3	34.8	63.0	51.6
δ-LCS$_c$	3	59.4	64.6	60.8	**55.4**	23.5	34.3	**63.6**	51.7

Table 15: BLEU score (en-fr): BLOOM inferred on filter-free retrieval settings.