Automating Behavioral Testing in Machine Translation

Javier Ferrando[◊*] **Matthias Sperber**[†] **Hendra Setiawan**[†]
Dominic Telaar[†] **Saša Hasan**[†]
[◊]Universitat Politècnica de Catalunya
[†]Apple
javier.ferrando.monsonis@upc.edu,sperber@apple.com

Abstract

Behavioral testing in NLP allows fine-grained evaluation of systems by examining their linguistic capabilities through the analysis of input-output behavior. Unfortunately, existing work on behavioral testing in Machine Translation (MT) is currently restricted to largely handcrafted tests covering a limited range of capabilities and languages. To address this limitation, we propose to use Large Language Models (LLMs) to generate a diverse set of source sentences tailored to test the behavior of MT models in a range of situations. We can then verify whether the MT model exhibits the expected behavior through matching candidate sets that are also generated using LLMs. Our approach aims to make behavioral testing of MT systems practical while requiring only minimal human effort. In our experiments, we apply our proposed evaluation framework to assess multiple available MT systems, revealing that while in general pass-rates follow the trends observable from traditional accuracy-based metrics, our method was able to uncover several important differences and potential bugs that go unnoticed when relying only on accuracy.[1]

1 Introduction

Automatic evaluation metrics such as BLEU (Papineni et al., 2002) and COMET (Rei et al., 2020) are the primary means of measuring the translation quality of MT systems. Researchers and practitioners rely on them for comparing systems, detecting regressions, and making deployment decisions. This poses an important concern: such metrics typically aggregate the performance of systems across a set of sentences into single scores. Unfortunately, these metrics by design tend to overlook specific infrequent but important error cases, making it difficult to reliably detect such issues in practice.

Property	**Translation Errors**
Integers	7000000 → 70.000.000
Decimals	500.75 → 500.75
Large Numbers	1.366 billion → 1.366 Milliarden
Idioms	ins and outs → Ins und Outs
Currencies	BRL → RL
Physical Units	miles → km
Web Terms	www.onlinegrocery.com → www.onlineegrocery.com
...	...

Table 1: Subset of linguistic properties tested with our proposed method, and examples (source → translation) of translation errors found in En→De MT models.

Behavioral testing, originally developed as a type of software testing (Beizer and Wiley, 1996), has been proposed as an approach that can alleviate such kinds of problems in natural language processing (Ribeiro et al., 2020). Behavioral tests focus on assessing a system's fine-grained linguistic capabilities by validating input-output behavior in a controlled fashion.

Table 1 shows examples of typical issues of MT systems that could be covered by behavioral tests. We argue that the availability of a comprehensive behavioral test suite for MT would be of high practical value: It would allow understanding how exactly two MT models differ, or to block an MT system from being deployed if a passing threshold for a certain linguistic capability is not met.

However, there are currently two major limitations that arise when attempting to apply behavioral testing to MT. First of all, behavioral testing was originally designed for evaluating systems characterized by a relatively small output space. For instance, Ribeiro et al. (2020) investigate sentiment classification, duplicate question detection, and machine comprehension. In contrast, the output space of MT systems grows exponentially as tokens are generated. Secondly, behavioral testing often requires rigid templates to create examples and their

[*] Work done during an internship at Apple.

[1]Prompts and generated data are available at https://github.com/apple/ml-behavioral-testing-for-mt.

Proceedings of the Eighth Conference on Machine Translation (WMT), pages 1014–1030
December 6–7, 2023. ©2023 Association for Computational Linguistics

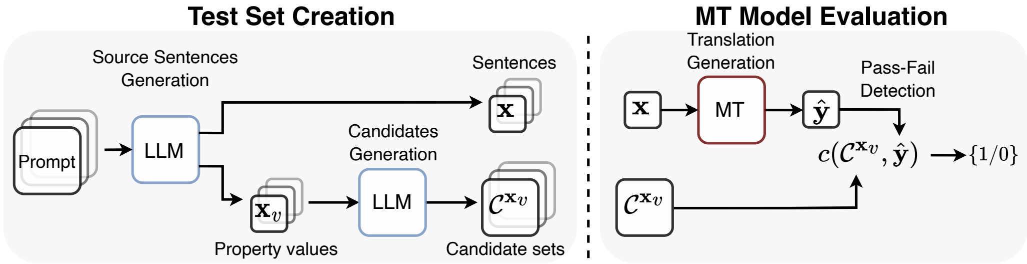

Figure 1: Pipeline of the proposed approach. Left: For each property type, a test set is created via a LLM, composed of source sentences $\mathbf{x}$ with property values $\mathbf{x}_v$ (§3). Subsequently, a candidate set of valid translations of each property value $\mathcal{C}^{\mathbf{x}_v}$ is generated (§4). Right: During evaluation, the translation $\hat{\mathbf{y}}$ generated by an MT model is compared against the candidate sets, and a pass-fail decision is made (§5).

corresponding labels, which involves a costly human effort to develop and expand to additional use cases. Otherwise, the diversity of sentences in the resulting test suite is too limited.

Several recent works have partially addressed these limitations. For example, Wang et al. (2021); Raunak et al. (2022) propose MT-specific test sets which include the ability to handle large output space. Yang et al. (2022) address the limitation of rigid templates. To the best of our knowledge, no prior work has addressed both limitations for MT.

In this study, we aim to bridge this gap by leveraging LLMs with in-context learning to automate the creation of behavioral tests in MT for the first time. Our main contributions are as follows:

- We use LLMs to automate the generation of a diverse set of source sentences for behavioral testing. Sentences are generated to exhibit the specific language property that is being tested.
- We verify whether an MT system's output contains an accurate translation of the language property that is being tested. To this end, we propose using LLMs to generate candidate sets of ground-truth translations of the property values in cases where exhaustive candidate sets are plausible. Otherwise, we generate contrastive candidates and evaluate via semantic similarity measures.
- We present an evaluation framework to robustly compute pass rates of MT models across various language properties, and show results for widely used open-source models on three language pairs.

2 Behavioral Testing for MT

Behavioral testing, as proposed by Ribeiro et al. (2020), uses input-output pairs tailored to evaluate a model capability to correctly handle certain language properties. The goal is to complement traditional aggregated accuracy scores, which, while useful by themselves, often fail to capture long-tail phenomena. In practice, manual inspection of system outputs is often crucial to make up for this shortcoming. Automated behavioral testing provides a more reliable and less cumbersome alternative that can reduce or eliminate the need for manual inspection, provided that a sufficient range of language properties is tested. Test results are presented in the form of a table of pass rates (one pass rate for each tested property) that is informative to decide on consequent steps, e.g. whether bugs must be addressed before deployment. Note that the creation of a *sufficiently comprehensive* behavioral test suite depends crucially on whether its creation can be automated to a high degree, which is also our main design goal in this work.

We are particularly interested in a specific type of behavioral tests, *minimal functionality tests* (MFTs) (Ribeiro et al., 2020).[2] In the context of MT, an MFT measures a model's ability to translate particular property values that appear naturally embedded in some given source sentences.

Figure 1 illustrates our proposed framework. First, a source sentence $\mathbf{x} = \{\boldsymbol{x}_1 \cdots, \boldsymbol{x}_{|\mathbf{x}|}\}$ that contains a tagged property value $\mathbf{x}_v \subseteq \mathbf{x}$ is generated (§3). For instance, if our test property is physical unit translation, we might have $\mathbf{x}$=*"I ran 3 miles."* and $\mathbf{x}_v$=*miles*. A main challenge comes from the fact that there is a potentially large space of correct translations. However, note that by design MFTs only need to check whether the property under test is translated correctly, while un-

[2] Ribeiro et al. (2020) also propose *directional* and *invariance* tests which check how model outputs change under certain input perturbance, but these appear less applicable to MT given the potentially large space of correct translations.

```
You are an assistant that generates sentences where only appears one B = {property}.
Don't be repetitive, change the topic and B between sentences. Write every B inside [].
B must happen only once in each sentence and can only contain {property}.

Write 3 examples.

- {Source sentence demonstration #1}
- {Source sentence demonstration #2}
- {Source sentence demonstration #3}

Now write 10 more diverse sentences itemizing them with '-':
```

Figure 2: General template of the prompt used for generating batches of source sentences.

related translation errors should be ignored. In many cases, this reduces the space of correct translations to a manageable size. We therefore propose to automatically generate a candidate set $\mathcal{C}^{\mathbf{x}_v}$ (either exhaustive or contrastive; see §4) and then apply a pass-fail detector that uses either string matching or semantic similarity measures (§5). In our example, we might generate an exhaustive candidate set $\mathcal{C}^{\mathbf{x}_v}=\{Meilen, mi\}$ for the case of translating into German. We now aim to evaluate an MT model $f : \mathbf{x} \mapsto \hat{\mathbf{y}}$. To do so, we compare $\hat{\mathbf{y}}$ against $\mathcal{C}^{\mathbf{x}_v}$. A correct translation $\hat{\mathbf{y}}$= *"Ich lief 3 Meilen."* would match the candidate set and therefore pass the test, while a typical incorrect translation $\hat{\mathbf{y}}$= *"Ich lief 3 km."* does not match the candidate set and therefore fails the test.

Given this general overview of our method, we now turn to a more precise description of each proposed step in the following sections.

3 Source Sentence Generation

To create source sentences for testing a certain language property, we pose several desiderata: Sentences should be *diverse* (e.g. not rely only on a handful of templates), *natural*, *numerous* enough to yield statistical significance, and *contain a property value* associated with our tested property.

Note that existing approaches often struggle to generate diverse test sets due to the reliance on hand-crafted templates (Wang et al., 2021). To overcome this shortcoming, we design a general template for prompting LLMs, in our case ChatGPT[3], OpenAI's model built on InstructGPT (Ouyang et al., 2022). This allows us to generate diverse source language sentences that contain property values suitable for testing different capabilities (see prompt[4] in Figure 2). We instantiate the prompt once for every language property that we wish to include in our test suite.

Candidates Examples
kilometers → kilómetros, km
watts → vatios, W
meters per second → metros por segundo, m/s

Table 2: Examples of En→Es set of candidates generated by ChatGPT.

To simplify the later verification step, we generate sentences that contain exactly one such property value $\mathbf{x}_v$.[5] We generate source sentences with brackets around the property value for easy parsing. A possible test sentence for the property of translating decimal numbers might look as follows:

$$\text{The company received [} \overbrace{4200.4}^{\text{property value}} \text{]€.} \tag{1}$$

Note that brackets are removed before passing the sentence to the MT model.

We apply basic filters to remove duplicated sentences, examples with more than one property value, or those composed of more than one sentence. We repeatedly feed the same prompt to the LLM, and stop the generation process when reaching 1,000 sentences after filtering. Our experiments (§9) indicate that ChatGPT is able to generate sentences of adequate quality and diversity.

4 Candidates Generation

Next, in order to be able to verify whether an MT system correctly translated the property value in the source sentence, we automatically generate valid translation candidate sets for each property value. For some properties, such as number translation, we create exhaustive or near-exhaustive candidate sets. For other properties where the number of valid translations would be too big to do so, we instead

[3] `gpt-3.5-turbo` API accessed on May 2023.

[4] We set temperature=0.9, presence_penalty=2.

[5] For some types of properties, multiple property values may be more appropriate. This is left for future work.

create contrastive candidate pairs that demonstrate desired and undesired behavior. Note that candidate sets only need to be created once and can then be re-used for every tested system.

4.1 Near-Exhaustive Candidate Sets

In this approach, we follow Raunak et al. (2022) in creating a set of all valid translations of each property value in the test (see example in Table 2). However, instead of manually designing candidate sets, we propose using the in-context learning (Brown et al., 2020) and multilingual capabilities of instruction-tuned LLMs (Wei et al., 2022) to accomplish the task. For each property value $\mathbf{x}_v$, we generate a set of translation candidates $\mathcal{C}^{\mathbf{x}_v}$ with ChatGPT (`gpt-3.5-turbo`) (see prompt[6] in Figure 3). We tried to design demonstrations to encompass both correctness and completeness, including possible inflections. An example of demonstrations used for the currencies test can be seen in Appendix B. Note that while we aim for completeness, i.e. all valid translations should be included in the candidate set, in practice we found that some rare translation choices may not be included in the automatically generated candidate sets. However, this will not impact pass-rates much because by nature rare translation choices appear in the MT system's output only in rare situations. In §9 we perform a human assessment of the reliability of the generated candidate sets.

4.2 Contrastive Candidate Pairs

Some property values can span multiple words on the source side, potentially increasing the number of valid translations drastically. An example is idiomatic expressions, where there is an increased risk that the candidate set cannot exhaust all possibilities. To mitigate this issue, we propose using *contrastive* candidate sets an alternative approach.

Given a source property value, we generate a *contrastive* candidate set $\mathcal{C}^{\mathbf{x}_v}_{\text{contra}}$ formed by a correct translation $c^{\mathbf{x}_v}_{\text{corr}}$, and a foil (incorrect) translation $c^{\mathbf{x}_v}_{\text{foil}}$. Appendix C shows an example prompt. Intuitively, an MT model should pass the test sentence if its translation is closer to $c^{\mathbf{x}_v}_{\text{corr}}$ than it is to $c^{\mathbf{x}_v}_{\text{foil}}$.

5 Pass-Fail Detector

Equipped with these candidate sets, we now wish to mark every MT-translated sentence as either *pass* or *fail*. Depending on whether near-exhaustive or contrastive candidate pairs are used, we design pass-fail detectors based on string matching or semantic similarity, respectively.

As it is our goal to design tests that target specific language properties, our pass-fail detectors should only detect cases where the property value under the test is translated incorrectly. Unrelated translation errors should not cause a sentence to be marked as incorrect.[7]

Algorithm 1: Similarity score between translation and contrastive candidate.

Input: $\hat{\mathbf{y}}$: model translation; c: candidate translation; e: encoder
Output: $\text{max\_sim}(\hat{\mathbf{y}}, c)$

$\text{max\_sim} \leftarrow -\infty$
$n \leftarrow |c|$
$\mathcal{G}_{\hat{\mathbf{y}}} \leftarrow n\text{-gram}(\hat{\mathbf{y}}, n)$
$\boldsymbol{c}_{\text{emb}} \leftarrow e(c)$
for $\mathbf{g} \in \mathcal{G}_{\hat{\mathbf{y}}}$ **do**
  $\boldsymbol{g}_{\text{emb}} \leftarrow e(\mathbf{g})$
  if $sim(\boldsymbol{g}_{emb}, \boldsymbol{c}_{emb}) > max\_sim$ **then**
    $\text{max\_sim} \leftarrow \text{sim}(\boldsymbol{g}_{\text{emb}}, \boldsymbol{c}_{\text{emb}})$
return max_sim

5.1 String Matching for Near-Exhaustive Candidate Sets

For the near-exhaustive candidate sets, we define a pass-fail function $c(\hat{\mathbf{y}}, \mathcal{C}^{\mathbf{x}_v}) \in \{0, 1\}$ that takes the model's translation $\hat{\mathbf{y}}$, and the candidates set $\mathcal{C}^{\mathbf{x}_v}$, and returns 1 (pass) if $\hat{\mathbf{y}}$ has a valid translation of the property value, i.e. if it has an element in $\mathcal{C}^{\mathbf{x}_v}$, and 0 (fail) otherwise:

$$c(\hat{\mathbf{y}}, \mathcal{C}^{\mathbf{x}_v}) = \begin{cases} 1 & \text{if } \hat{\mathbf{y}} \cap \mathcal{C}^{\mathbf{x}_v} \neq \emptyset \\ 0 & \text{otherwise.} \end{cases} \tag{2}$$

Specifically, we consider as pass an exact case-insensitive substring matching. Following Example 1, where $\mathbf{x}_v = 4200.4$, if we are evaluating the En→De decimal numbers translation capabilities of the model, we would consider the model passes the test if it outputs '4200,4', or '4.200,4'.[8]

[6] We use the same set of parameters as for the source sentence generation.

[7] For our purposes, we do not consider whether the translated property is placed at the correct position in the target sentence, but only whether it is correct when considered in isolation. We argue that errors related to fluency and reordering are better evaluated through established accuracy-based metrics.

[8] Note that this involves a design decision: The test case writers must make a decision whether or not the added decimal point is acceptable for their particular use cases.

```
You are a {source_lang}-{target_lang} translator. Given a {property}, write as many valid {target_lang}
translations as you can. Use "|" to separate between valid translations.
Write "NA" if unable to accomplish the task.

{Source property demonstration #1}
{Candidates set source property demonstration #1}
{Source property #2}
{Candidates set source property demonstration #2}
{Source property #3}
{Candidates set source property demonstration #3}

{Source property}
```

Figure 3: General template of the prompt used for generating near-exhaustive sets of candidate translations.

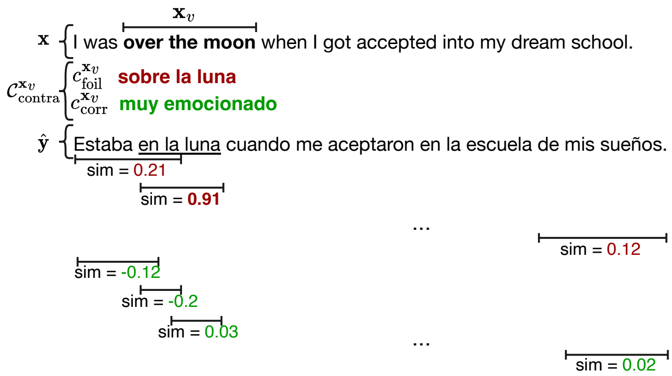

Figure 4: Example of the Contrastive Candidate Pairs approach, where `sim` indicates the semantic similarity between the correct candidate ('muy emocionado') and the 2-grams (in green), and the foil candidate ('sobre la luna') and the 3-grams of the MT translation (in red).

5.2 Semantic Similarity for Contrastive Candidate Pairs

For measuring the closeness of the property value translation to the contrastive candidates, we propose relying on the semantic similarity of word sequences representations extracted by a multilingual encoder (Reimers and Gurevych, 2019, 2020).[9] However, directly measuring the similarity between the translation of the property value and the candidate sets may be unreliable since they may differ in length and the location of the translation is unknown due to lack of word-level alignment. Instead, we propose that, for each candidate $c_{\text{corr}}^{\mathbf{x}_v}$ or $c_{\text{foil}}^{\mathbf{x}_v}$, we split the model's translation into n-grams, where n is the number of words of the current candidate. Then, we measure the similarity between each of the n-grams and the candidates.

Given a translation and the *contrastive* candidate set $\mathcal{C}_{\text{contra}}^{\mathbf{x}_v}$ formed by the correct and foil candidates, we define the pass-fail function as:

$$c(\hat{\mathbf{y}}, \mathcal{C}_{\text{contra}}^{\mathbf{x}_v}) = \begin{cases} 1 & \text{if } \text{max\_sim}(\hat{\mathbf{y}}, c_{\text{corr}}^{\mathbf{x}_v}) \geq \text{max\_sim}(\hat{\mathbf{y}}, c_{\text{foil}}^{\mathbf{x}_v}) \\ 0 & \text{otherwise.} \end{cases} \tag{3}$$

Algorithm 1 formalizes the computation of max_sim function, Figure 4 shows an example.

6 Evaluation Metrics

Having established pass-fail detection for individual sentences, the final step is to compute aggregated *pass rates* across test sets. Appealingly, pass rates are naturally expressed as percentages, making them intuitive to interpret.

6.1 Macro Pass Rate

Let us assume that we have computed pass-fail results across a behavioral test set consisting of N test cases (sentences). From a statistical viewpoint, we have access to a sample $\mathcal{X} = \{c(\hat{\mathbf{y}}^n, \mathcal{C}^{\mathbf{x}_v^n})\}_{n=1}^N$, drawn from some unknown distribution over test cases, F. The expectation of the true pass rate can be computed as follows:

$$\text{PR}^{(\mathcal{X})} = \frac{1}{N} \sum_{n}^{N} c(\hat{\mathbf{y}}^n, \mathcal{C}^{\mathbf{x}_v^n}) \tag{4}$$

[9]Employing LLMs is also possible but not explored here because it needs to be applied for every evaluated MT system, incurring higher computational costs.

Model	En→De			En→Es			En→Ja		
	spBLEU	ChrF	COMET-22	spBLEU	ChrF	COMET-22	spBLEU	ChrF	COMET-22
M2M 418M	31.08	57.22	79.49	25.33	51.26	80.63	23.57	32.22	84.84
M2M 1.2B	39.37	62.51	85.35	29.06	53.85	84.22	27.46	35.25	87.63
NLLB 600M	38.88	61.85	85.89	30.65	54.76	85.34	18.75	29.62	86.72
NLLB 3.3B	44.41	65.26	87.98	32.69	56.09	86.39	20.76	32.5	88.12
OPUS MT (Bil)	40.96	63.49	84.61	30.57	54.97	84.9	-	-	-
WMT21 (En-X)	**49.38**	**68.94**	88.76	-	-	-	39.89	44.95	91.95
Commercial system	49.34	68.84	**89.34**	**34.43**	**57.58**	**86.92**	**41.05**	**47.06**	**92.19**

Table 3: Translation scores of the different models used in FLORES-200 devtest set.

One issue that arises in practice is that property values themselves follow a long tail pattern: Certain values appear relatively frequently, while many other values appear only once across the generated test set. This can make pass rates overly sensitive to whether models happen to perform well for these particular values. To mitigate this issue, we assume a generative story in which property values are drawn from a uniform distribution, and consequently compute the expected pass rate as the macro average across property values:

$$\mathrm{MPR}^{(\mathcal{X})} = \frac{1}{|\mathcal{V}|}\sum_{v\in\mathcal{V}}\frac{1}{N_v}\sum_{i}^{N_v} c(\hat{\mathbf{y}}^i, \mathcal{C}^{\mathbf{x}_v^i}) \quad (5)$$

where $\mathcal{V}$ refers to the set of distinct property values, and N_v to the number of examples associated with each specific property value.

6.2 Confidence Intervals

Although previous work performing behavioral testing for MT shows point estimate scores, confidence intervals provide a more reliable approach to statistical analysis, as they quantify the uncertainty associated with that estimate, and ensure the sample size is large enough. To compute confidence intervals for our estimator MPR we use the Bootstrap method (Efron, 1979), which performs sampling with replacement from $\mathcal{X}$, generating K resamples $\{\mathcal{Y}^1, \cdots, \mathcal{Y}^K\}$, from which we compute their corresponding macro pass rates $\{\mathrm{MPR}^{(\mathcal{Y}^1)}, \cdots, \mathrm{MPR}^{(\mathcal{Y}^K)}\}$ to construct the bootstrap distribution $\mathrm{MPR}_{\mathrm{boot}}$. Assuming the distribution of $\mathcal{X}$ is a reasonable approximation of the population distribution F, confidence intervals can be derived from $\mathrm{MPR}_{\mathrm{boot}}$. For that purpose, we compute the percentile bootstrap interval for $\alpha = 0.05$ provided by SciPy library (Virtanen et al., 2020).

6.3 Paired Bootstrap

The paired bootstrap is a statistical resampling technique used to assess the uncertainty and make inferences about the difference between two samples. Paired bootstrap allows us to compare the property's sample of passes/fails for two different models (Koehn, 2004). By following the resampling process outlined in the previous section, if a model consistently outperforms the other in 95% of the iterations, we can assert with 95% statistical significance that it is superior.

7 Properties to Test

We design a number of tests and use our proposed framework to evaluate MT models in multiple properties. The chosen properties, also studied in the literature (Wang et al., 2021; Raunak et al., 2022), have two important qualities that make them useful for evaluating translation systems: vital for producing high-quality translations, yet posing a challenge when assessing through conventional evaluation metrics.

Numbers. We conduct independent assessments for integers (e.g. 1887), decimals (e.g. 154.32), and large numbers (e.g. 200 billion). Large numbers have the format "integer/decimal million/billion/trillion". We create near-exhaustive candidate sets of valid number translations and check if the translation matches any candidate.

Physical Units. We build near-exhaustive candidate sets for evaluating the translations of diverse units including those related to weight, length, time, or temperature *inter alia* (e.g. inches). Translations are evaluated by string matching.

Emojis, Names, and Web Terms. Via string matching we check whether the translated text retains the same property instantiation found in the source text. Candidate sets for these tests are thus considered to be exhaustive.

Currencies. We consider currencies appearing in the ISO code format (e.g. EUR). Near exhaustive candidate sets are built allowing translations into the same ISO code, variations of the currency name or its symbol (e.g. for En→Es: EUR/euro/euros/€), then a string matching pass-fail detection is employed.

Idioms. Idiomatic expressions pose significant challenges for MT systems due to their non-literal nature and potential large sequence length. We use idioms as a test bench for the use of contrastive candidate pairs (incorrect literal translation candidate vs. correct meaning translation) and semantic similarity detection procedure.

8 Models Comparison

In this section, we introduce the tested models and present results obtained via standard metrics as well as our proposed framework.

8.1 Experimental Setup

We test widely-used open-source MT models, as well as a commercial system. We aim to select models that perform very strongly, while also differing in some important aspects (e.g. bilingual vs. multilingual).

In the multilingual domain, we experiment with the 600M and the 3.3B parameters models of No Language Left Behind project (NLLB) (Team et al., 2022), and the Many-to-Many (MLM-100) family of Multilingual models (Fan et al., 2021) (418M and 1.2B parameters models). Additionally, we evaluate the WMT21: multilingual (7 En→X directions) 4.7B dense model (Tran et al., 2021), part of Meta's WMT-21 News Translation task participation (Barrault et al., 2021). We also assess OPUS-MT (Tiedemann and Thottingal, 2020) En→Es and En→De bilingual models trained on OPUS dataset (Tiedemann, 2012). Lastly, we included results from an anonymous commercial system.

Besides our proposed metrics, we also evaluate the models on FLORES-200 (Team et al., 2022) in En→De, En→Es, and En→Ja via string-based metrics spBLEU[10] (Papineni et al., 2002) and ChrF[11] (Popović, 2015) as implemented in SACREBLEU (Post, 2018), as well as the neural-based metric COMET-22[12] (Rei et al., 2020).

[10] SACREBLEU signature: `nrefs:1|case:mixed|eff:no|tok:flores101|smooth:exp|version:2.3.1`

[11] SACREBLEU signature: `nrefs:1|case:mixed|eff:yes|nc:6|nw:0|space:no|version:2.3.1`

Model	Source Sentence	Translation
OPUS MT (Bil)	The article I read on `www.scientificjournal.org` was very informative.	El artículo que leí en `www.cientificojournal.org` fue muy informativo.
Commercial system	... our town's population was counted as 12,577.	... población de nuestra ciudad se contabilizó en 12,577.

Table 4: Examples flagged as failed translations.

8.2 General Translation Accuracy

We first measure general translation performance across language pairs for standard reference-based metrics (Table 3). The commercial system performs best across the board, followed by the WMT21 model. In the following sections, we dive deeper into the different capabilities.

8.3 Behavioral Tests Results

As an illustrative example, macro pass rate confidence intervals across property types and models for the En→De direction are presented in Figure 5. The complete results can be found in Appendix E.

Commercial system is most consistent across properties. This is especially true for emoji translations, where open-source models lack most emojis in their vocabulary. However, it is noteworthy that its performance is subpar in the context of En→Es integers and En→Ja large numbers. After manual inspection (see examples in Table 4), we attribute the lower integers translation performance to the fact that it uses the comma as the thousands separator. Note that this behavior can be acceptable depending on the country; behavioral tests must be designed to reflect the intended behavior.

Bilingual models struggle with web terms. Although the multilingual models mostly manage to preserve web terms without alteration, both tested bilingual models (for En→Es and En→De) underperform in that property (see Figure 5 and Figure 6 top). Most fail cases contain Spanish words inside the translated web terms (Table 4). We hypothesize that this occurs because they are trained to exclusively translate into Spanish, which consequently hinders their ability to generate content in other

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

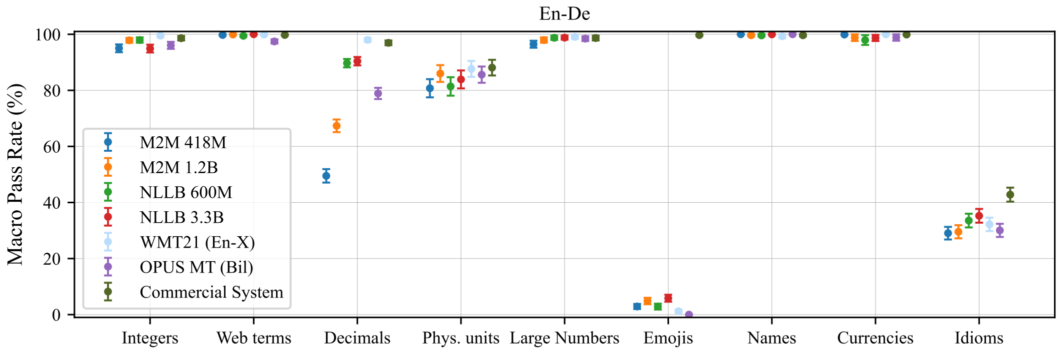

Figure 5: En→De macro pass rates and confidence intervals across tested systems.

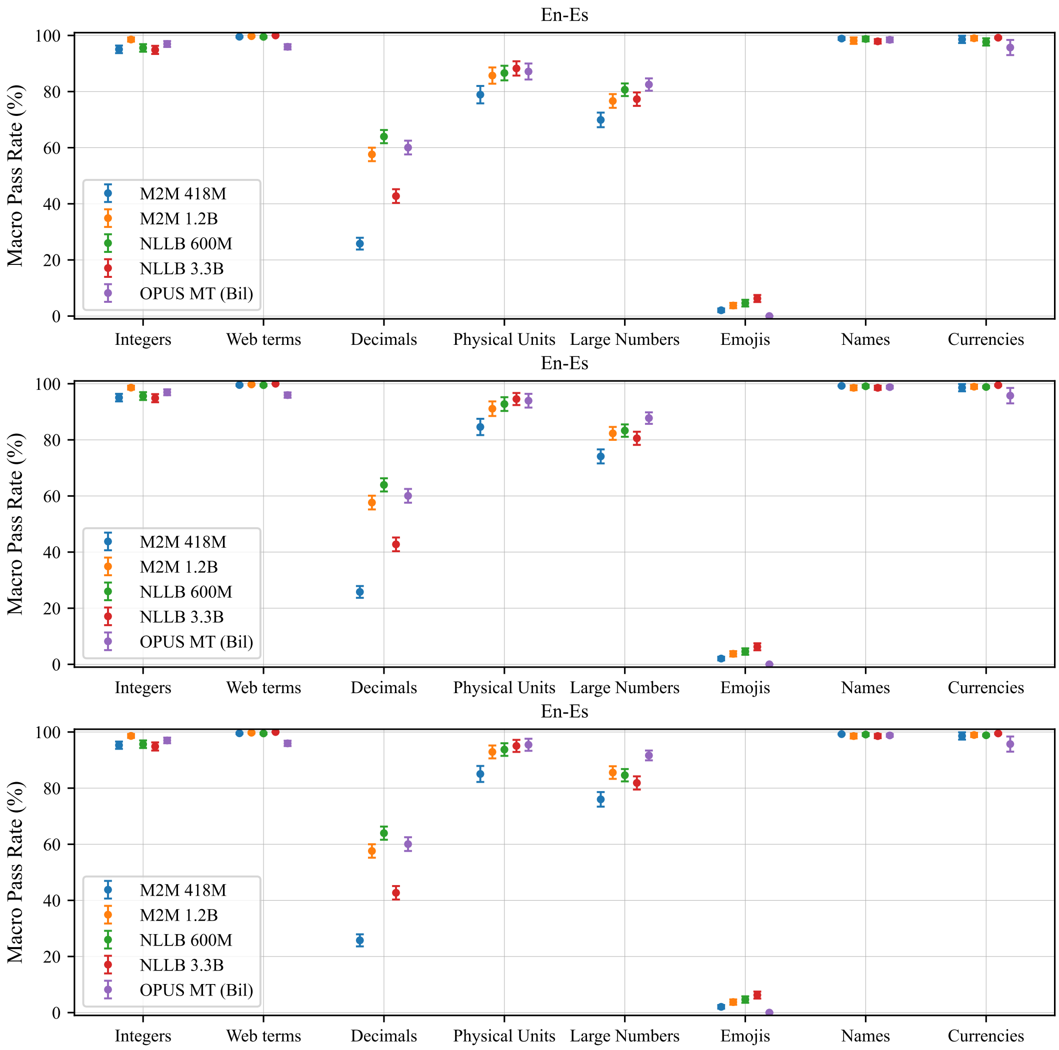

Figure 6: From top to bottom, En→Es Confidence Intervals after each annotation iteration (see §9).

languages, and is therefore an intrinsic limitation of bilingual models.

Scaling models help increase capabilities. In most of the settings, scaling the model of the same family shows increased performance, for instance, physical units and idioms in Figure 5. However, there are some counter-examples, like in the case of En→Ja decimals and integers tests.

WMT21 is the strongest open-source model. The WMT21 model consistently exhibits superior performance compared to other open-source models in both En→De and En→Ja tests. In Table 5 we show how paired bootstrap enables model comparison, revealing that WMT21 outperforms other

Model A	Model B	Winner	p-value
M2M 418M	M2M 1.2B	M2M 1.2B	0.0
M2M 418M	NLLB 600M	NLLB 600M	0.0
M2M 418M	NLLB 3.3B	M2M 418M	0.476
M2M 418M	WMT21 (En-X)	WMT21 (En-X)	0.0
M2M 418M	OPUS MT (Bil)	OPUS MT (Bil)	0.106
M2M 418M	Commercial system	Commercial system	0.0
M2M 1.2B	NLLB 600M	NLLB 600M	0.461
M2M 1.2B	NLLB 3.3B	M2M 1.2B	0.0
M2M 1.2B	WMT21 (En-X)	WMT21 (En-X)	0.001
M2M 1.2B	OPUS MT (Bil)	M2M 1.2B	0.004
M2M 1.2B	Commercial system	Commercial system	0.102
NLLB 600M	NLLB 3.3B	NLLB 600M	0.0
NLLB 600M	WMT21 (En-X)	WMT21 (En-X)	0.003
NLLB 600M	OPUS MT (Bil)	NLLB 600M	0.005
NLLB 600M	Commercial system	Commercial system	0.142
NLLB 3.3B	WMT21 (En-X)	WMT21 (En-X)	0.0
NLLB 3.3B	OPUS MT (Bil)	OPUS MT (Bil)	0.118
NLLB 3.3B	Commercial system	Commercial system	0.0
WMT21 (En-X)	OPUS MT (Bil)	WMT21 (En-X)	0.0
WMT21 (En-X)	Commercial system	WMT21 (En-X)	0.024
OPUS MT (Bil)	Commercial system	Commercial system	0.0

Table 5: Paired Bootstrap En→De Integers test results. We make a 95% statistically significant conclusion that the WMT21 system is better than the rest of the models.

models in the integers En→De test.

Idioms. Results for the Idioms property test are presented in Appendix D. The ability to translate idioms is generally low (i.e. overly literal), in accordance with recent findings (Dankers et al., 2022). It is worth noting that results are similar in the three language directions, with the commercial system and NLLB 3.3B showing comparable performance.

9 Reliability of the Proposed Approach

To assess the reliability of the proposed approach, in this section we analyze the robustness of source sentence generation and pass-fail detection.

9.1 Analysis of Source Sentence Generation

One potential concern with the proposed method is whether the generated source sentences are diverse enough and do not become repetitive after a few rounds of generation.[13] A standard method for quantifying the diversity in a corpus is *distinct n-grams* (Li et al., 2016), which computes the ratio of unique n-grams to the total number of n-grams present. In our case, we are interested in assessing the diversity of each generated source sentence compared to the previous generations. To that end, we propose a metric to measure this aspect. Given the set of unique n-grams generated up to sentence $\mathbf{x}_t$ ($\mathcal{G}^n_{\mathbf{x}_{<t}}$), we measure the proportion of unique

[13]The *naturalness* of outputs, another potential concern, has been extensively dealt with elsewhere (Ouyang et al., 2022).

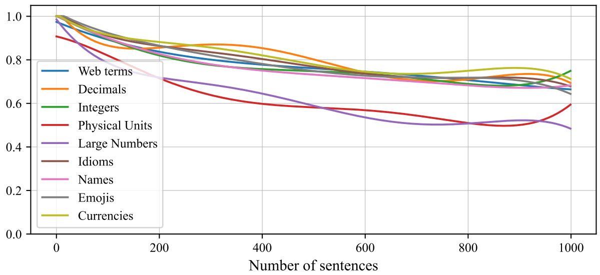

Figure 7: 3-gram diversity scores ($\text{div}_3(\mathbf{x}_t)$) along generation steps across different properties.

Property	Sentences kept	Unique values
Web Terms	79.3%	92.5%
Decimals	74.1%	76.9%
Integers	62.1%	39.3%
Physical Units	83.3%	15.9%
Large Numbers	66.1%	37.1%
Idioms	83.8%	69.0%
Names	86.1%	17.9%
Emojis	88.5%	29.7%
Currencies	66.8%	5.2%

Table 6: Percentage of source sentences that pass filtering, and percentage of filtered sentences that introduce a new property value.

n-grams in each newly generated sentence ($\mathcal{G}^n_{\mathbf{x}_t}$) that are not present in $\mathcal{G}^n_{\mathbf{x}_{<t}}$:

$$\text{div}_n(\mathbf{x}_t) = \frac{\mathcal{G}^n_{\mathbf{x}_t} \setminus \mathcal{G}^n_{\mathbf{x}_{<t}}}{\mathcal{G}^n_{\mathbf{x}_t}} \quad (6)$$

Figure 7 shows 3-gram diversity along 1000 generated sentences after fitting a polynomial regression. We observe that the diversity drop is mild even after 500 sentences, where for most of the tests, 60% of newly generated 3-grams are novel.

Furthermore, we observe that the sentence generator produces sentences that comply with instructions, indicated by the high proportion of the original sentences that pass filtering. In the majority of cases, over 70% of the LLM-generated sentences successfully pass the filtering steps outlined in §3, as seen in Table 6 (middle column). The right column shows the percentage of unique values, which naturally vary strongly depending on the property.

9.2 Analysis of Pass-Fail Detection

The reliability of the proposed pass-fail detection depends mainly on whether candidate sets are (1) complete and (2) do not contain wrong candidates.

We analyze this by sampling 100 random test cases that were marked as *pass* (positives), and an-

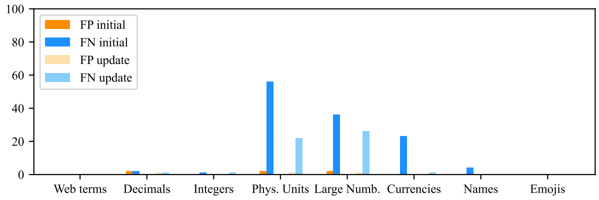

Figure 8: Error rates detected in two rounds of annotations on En→Es.

other 100 examples marked as *fail* (negatives). We manually annotate whether test results were correct or incorrect. Figure 8 shows false positives and false negatives (FP initial and FN initial). We observe that while for most properties these were low, for some test cases (namely physical units, large numbers, currencies) there were a significant number of FNs, which would lead to underestimated pass rates. We argue that erring on the side of FNs is generally preferable, because it prevents us from overestimating the strength of models, and because it would trigger a debugging effort which would quickly surface issues stemming from FNs.

To obtain more accurate pass-rates for all properties, we can manually remove candidates causing a FP and add missing candidates producing a FN. We do this for the test cases analyzed above, and then draw another random sample from both *pass* and *fail* categories. Figure 8 shows that the updated FPs and FNs are now negligible.

While in our experience, human intervention as outlined above is only a minor effort, the issue remains as to whether systems can be compared to one another without the need for human intervention, even in the presence of existing FNs. To understand this better, we plot macro pass rates with confidence intervals across annotation iterations in Figure 6. As expected, for physical units, large numbers, and currencies, pass rates move upwards. However, the effect is general across models, suggesting that relative ordering between models can be reasonably approximated in the initial attempt, i.e. without human intervention.

In addition, we assess the pass-fail detection of idioms. Given that the decision is made via semantic similarity for contrasting pairs, addressing issues in the candidate sets is more challenging. Consequently, we conducted a single evaluation iteration with 100 *pass*/*fail* examples, respectively, on two language pairs. For En→De, we observed 59 FPs / 16 FNs; En→Es had 50 FPs / 11 FNs. We hypothesize high FPs are caused by idiom and its figurative meaning being present within the source sentence, interfering with the n-grams comparison. We leave further investigation for future research.

10 Related Work

Recent works have applied behavioral testing for evaluating machine translation systems. Wang et al. (2021) designed tests for numerical translation capabilities by relying on fixed templates for source sentence generation. Raunak et al. (2022) proposed SALTED, a set of manually designed error detectors that are applied to millions of sentences from standard datasets. Beyond behavioral testing, a large number of *challenge sets* have been developed for machine translation (Popović and Castilho, 2019). Although useful, most of these evaluation tools require major human efforts for creation, evaluation, or expanding to other languages. Although there have been attempts to automatize the creation of behavioral tests (Yang et al., 2022), this has been limited to simple NLP tasks.

Our work also relates to the use of LLMs as evaluators for Machine Translation systems (Kocmi and Federmann, 2023), as well as for text generation in a broader sense (Liu et al., 2023; Xu et al., 2023), which extend the growing body of research on multi-dimensional text generation evaluation (Zhong et al., 2022; Yuan et al., 2021).

Behavioral testing aims to evaluate the behavior of systems under realistic conditions, contrasting it from the literature on adversarial data generation (Belinkov and Bisk, 2018; Zhang et al., 2021).

11 Conclusions

In this work, we have presented a method that automates the creation of behavioral tests to perform fine-grained evaluation of MT systems capabilities. We use Large Language Models to generate source sentences composed of fragments of specific language properties (integers, web terms, etc.), as well as translations of these properties. For property types formed by multiple words, we further extend the proposed method into a contrastive setting and show its usefulness in evaluating idiomatic expressions. To the best of our knowledge, our research represents the first attempt to develop MT behavioral tests by leveraging LLMs. Finally, we apply the proposed framework to evaluate open-source models on three language pairs.

References

Loic Barrault, Ondrej Bojar, Fethi Bougares, Rajen Chatterjee, Marta R. Costa-jussa, Christian Federmann, Mark Fishel, Alexander Fraser, Markus Freitag, Yvette Graham, Roman Grundkiewicz, Paco Guzman, Barry Haddow, Matthias Huck, Antonio Jimeno Yepes, Philipp Koehn, Tom Kocmi, Andre Martins, Makoto Morishita, and Christof Monz, editors. 2021. *Proceedings of the Sixth Conference on Machine Translation*. Association for Computational Linguistics, Online.

B. Beizer and J. Wiley. 1996. Black box testing: Techniques for functional testing of software and systems. *IEEE Software*, 13(5):98–.

Yonatan Belinkov and Yonatan Bisk. 2018. Synthetic and natural noise both break neural machine translation. In *International Conference on Learning Representations*.

Tom B. Brown, Benjamin Mann, Nick Ryder, Melanie Subbiah, Jared Kaplan, Prafulla Dhariwal, Arvind Neelakantan, Pranav Shyam, Girish Sastry, Amanda Askell, Sandhini Agarwal, Ariel Herbert-Voss, Gretchen Krueger, Tom Henighan, Rewon Child, Aditya Ramesh, Daniel M. Ziegler, Jeffrey Wu, Clemens Winter, Christopher Hesse, Mark Chen, Eric Sigler, Mateusz Litwin, Scott Gray, Benjamin Chess, Jack Clark, Christopher Berner, Sam McCandlish, Alec Radford, Ilya Sutskever, and Dario Amodei. 2020. Language models are few-shot learners. In *Proceedings of the 34th International Conference on Neural Information Processing Systems*, NIPS'20, Red Hook, NY, USA. Curran Associates Inc.

Verna Dankers, Christopher Lucas, and Ivan Titov. 2022. Can transformer be too compositional? analysing idiom processing in neural machine translation. In *Proceedings of the 60th Annual Meeting of the Association for Computational Linguistics (Volume 1: Long Papers)*, pages 3608–3626, Dublin, Ireland. Association for Computational Linguistics.

B. Efron. 1979. Bootstrap methods: Another look at the jackknife. *The Annals of Statistics*, 7(1):1–26.

Angela Fan, Shruti Bhosale, Holger Schwenk, Zhiyi Ma, Ahmed El-Kishky, Siddharth Goyal, Mandeep Baines, Onur Celebi, Guillaume Wenzek, Vishrav Chaudhary, Naman Goyal, Tom Birch, Vitaliy Liptchinsky, Sergey Edunov, Edouard Grave, Michael Auli, and Armand Joulin. 2021. Beyond english-centric multilingual machine translation. *J. Mach. Learn. Res.*, 22(1).

Tom Kocmi and Christian Federmann. 2023. Large language models are state-of-the-art evaluators of translation quality.

Philipp Koehn. 2004. Statistical significance tests for machine translation evaluation. In *Proceedings of the 2004 Conference on Empirical Methods in Natural Language Processing*, pages 388–395, Barcelona, Spain. Association for Computational Linguistics.

Jiwei Li, Michel Galley, Chris Brockett, Jianfeng Gao, and Bill Dolan. 2016. A diversity-promoting objective function for neural conversation models. In *Proceedings of the 2016 Conference of the North American Chapter of the Association for Computational Linguistics: Human Language Technologies*, pages 110–119, San Diego, California. Association for Computational Linguistics.

Yang Liu, Dan Iter, Yichong Xu, Shuohang Wang, Ruochen Xu, and Chenguang Zhu. 2023. G-eval: Nlg evaluation using gpt-4 with better human alignment.

Long Ouyang, Jeff Wu, Xu Jiang, Diogo Almeida, Carroll L. Wainwright, Pamela Mishkin, Chong Zhang, Sandhini Agarwal, Katarina Slama, Alex Ray, John Schulman, Jacob Hilton, Fraser Kelton, Luke Miller, Maddie Simens, Amanda Askell, Peter Welinder, Paul Christiano, Jan Leike, and Ryan Lowe. 2022. Training language models to follow instructions with human feedback.

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.

Maja Popović. 2015. chrF: character n-gram F-score for automatic MT evaluation. In *Proceedings of the Tenth Workshop on Statistical Machine Translation*, pages 392–395, Lisbon, Portugal. Association for Computational Linguistics.

Maja Popović and Sheila Castilho. 2019. Challenge test sets for MT evaluation. In *Proceedings of Machine Translation Summit XVII: Tutorial Abstracts*, Dublin, Ireland. European Association for Machine Translation.

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.

Vikas Raunak, Matt Post, and Arul Menezes. 2022. SALTED: A framework for SAlient long-tail translation error detection. In *Findings of the Association for Computational Linguistics: EMNLP 2022*, pages 5163–5179, Abu Dhabi, United Arab Emirates. Association for Computational Linguistics.

Ricardo Rei, Craig Stewart, Ana C Farinha, and Alon Lavie. 2020. COMET: A neural framework for MT evaluation. In *Proceedings of the 2020 Conference on Empirical Methods in Natural Language Processing (EMNLP)*, pages 2685–2702, Online. Association for Computational Linguistics.

Nils Reimers and Iryna Gurevych. 2019. Sentence-BERT: Sentence embeddings using Siamese BERT-networks. In *Proceedings of the 2019 Conference on*

Empirical Methods in Natural Language Processing and the 9th International Joint Conference on Natural Language Processing (EMNLP-IJCNLP), pages 3982–3992, Hong Kong, China. Association for Computational Linguistics.

Nils Reimers and Iryna Gurevych. 2020. Making monolingual sentence embeddings multilingual using knowledge distillation. In *Proceedings of the 2020 Conference on Empirical Methods in Natural Language Processing (EMNLP)*, pages 4512–4525, Online. Association for Computational Linguistics.

Marco Tulio Ribeiro, Tongshuang Wu, Carlos Guestrin, and Sameer Singh. 2020. Beyond accuracy: Behavioral testing of NLP models with CheckList. In *Proceedings of the 58th Annual Meeting of the Association for Computational Linguistics*, pages 4902–4912, Online. Association for Computational Linguistics.

NLLB Team, Marta R. Costa-jussà, James Cross, Onur Çelebi, Maha Elbayad, Kenneth Heafield, Kevin Heffernan, Elahe Kalbassi, Janice Lam, Daniel Licht, Jean Maillard, Anna Sun, Skyler Wang, Guillaume Wenzek, Al Youngblood, Bapi Akula, Loic Barrault, Gabriel Mejia Gonzalez, Prangthip Hansanti, John Hoffman, Semarley Jarrett, Kaushik Ram Sadagopan, Dirk Rowe, Shannon Spruit, Chau Tran, Pierre Andrews, Necip Fazil Ayan, Shruti Bhosale, Sergey Edunov, Angela Fan, Cynthia Gao, Vedanuj Goswami, Francisco Guzmán, Philipp Koehn, Alexandre Mourachko, Christophe Ropers, Safiyyah Saleem, Holger Schwenk, and Jeff Wang. 2022. No language left behind: Scaling human-centered machine translation.

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).

Jörg Tiedemann and Santhosh Thottingal. 2020. OPUS-MT – building open translation services for the world. In *Proceedings of the 22nd Annual Conference of the European Association for Machine Translation*, pages 479–480, Lisboa, Portugal. European Association for Machine Translation.

Chau Tran, Shruti Bhosale, James Cross, Philipp Koehn, Sergey Edunov, and Angela Fan. 2021. Facebook AI's WMT21 news translation task submission. In *Proceedings of the Sixth Conference on Machine Translation*, pages 205–215, Online. Association for Computational Linguistics.

Pauli Virtanen, Ralf Gommers, Travis E. Oliphant, Matt Haberland, Tyler Reddy, David Cournapeau, Evgeni Burovski, Pearu Peterson, Warren Weckesser, Jonathan Bright, Stéfan J. van der Walt, Matthew Brett, Joshua Wilson, K. Jarrod Millman, Nikolay Mayorov, Andrew R. J. Nelson, Eric Jones, Robert Kern, Eric Larson, C J Carey, İlhan Polat, Yu Feng, Eric W. Moore, Jake VanderPlas, Denis Laxalde, Josef Perktold, Robert Cimrman, Ian Henriksen, E. A. Quintero, Charles R. Harris, Anne M. Archibald, Antônio H. Ribeiro, Fabian Pedregosa, Paul van Mulbregt, and SciPy 1.0 Contributors. 2020. SciPy 1.0: Fundamental Algorithms for Scientific Computing in Python. *Nature Methods*, 17:261–272.

Jun Wang, Chang Xu, Francisco Guzmán, Ahmed El-Kishky, Benjamin Rubinstein, and Trevor Cohn. 2021. As easy as 1, 2, 3: Behavioural testing of NMT systems for numerical translation. In *Findings of the Association for Computational Linguistics: ACL-IJCNLP 2021*, pages 4711–4717, Online. Association for Computational Linguistics.

Jason Wei, Maarten Bosma, Vincent Zhao, Kelvin Guu, Adams Wei Yu, Brian Lester, Nan Du, Andrew M. Dai, and Quoc V Le. 2022. Finetuned language models are zero-shot learners. In *International Conference on Learning Representations*.

Wenda Xu, Danqing Wang, Liangming Pan, Zhenqiao Song, Markus Freitag, William Yang Wang, and Lei Li. 2023. Instructscore: Towards explainable text generation evaluation with automatic feedback.

Guanqun Yang, Mirazul Haque, Qiaochu Song, Wei Yang, and Xueqing Liu. 2022. TestAug: A framework for augmenting capability-based NLP tests. In *Proceedings of the 29th International Conference on Computational Linguistics*, pages 3480–3495, Gyeongju, Republic of Korea. International Committee on Computational Linguistics.

Weizhe Yuan, Graham Neubig, and Pengfei Liu. 2021. Bartscore: Evaluating generated text as text generation. In *Advances in Neural Information Processing Systems*, volume 34, pages 27263–27277. Curran Associates, Inc.

Xinze Zhang, Junzhe Zhang, Zhenhua Chen, and Kun He. 2021. Crafting adversarial examples for neural 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 1: Long Papers)*, pages 1967–1977, Online. Association for Computational Linguistics.

Ming Zhong, Yang Liu, Da Yin, Yuning Mao, Yizhu Jiao, Pengfei Liu, Chenguang Zhu, Heng Ji, and Jiawei Han. 2022. Towards a unified multi-dimensional evaluator for text generation. In *Proceedings of the 2022 Conference on Empirical Methods in Natural Language Processing*, pages 2023–2038, Abu Dhabi, United Arab Emirates. Association for Computational Linguistics.

A Limitations

While the proposed evaluation framework seeks to address a broad spectrum of languages, the experiments conducted in this study are limited to three language pairs. Due to its reliance on the capacity of LLMs to produce high-quality candidate translations, we cannot guarantee accurate results when applied to language pairs involving a low-resource language using current LLMs. Moreover, the method is designed to work only on properties that appear as a continuous chunk of text in both source and target languages and are not scattered across a sentence.

B Example of Demonstrations for Exhaustive Candidate Set Generation

```
You are a {src_lang}-{tgt_lang} translator. Given a {property}, write as many valid
{target_lang} translations as you can. Use "|" to separate between valid translations.
Write "NA" if unable to accomplish the task.

EUR
€|EUR|Euro

GBP
£|GBP|Pfund|Pfund Sterling|britisches Pfund|Pound Sterling

USD
$|USD|Dollar|US Dollar|amerikanischer Dollar|amerikanische Dollar|US-Dollar

{Source property}
```

Figure 9: General template of the prompt used for generating a set of candidate translations.

C Example of Demonstrations for Contrastive Candidate Pairs Generation

Foil:

```
You are an {src_lang}-{tgt_lang} literal translator. Given a sequence of words,
you have to write only a literal translation. Use "|" to separate alternatives.
Write "NA" if unable to accomplish the task.

break a leg
brich dir ein Bein|breche dir ein Bein|breche dein Bein|breche dir dein Bein

hit the ground running
im Laufen hinfallen|beim Laufen hinfallen|beim Laufen auf den Boden knallen|beim Laufen
auf den Boden fallen

put all your eggs in one basket
alle Eier in einen Korb tun|alle Eier in einen Korb setzen|alle Eier in einen Korb legen

{Source property}
```

Correct:

```
You are an {src_lang_name}-{tgt_lang_name} translator of idiomatic expressions. Given an
idiomatic expression, you have to write the translation of the figurative meaning of the
idiomatic expression. Use "|" to separate alternatives. Write "NA" if unable to accomplish the task.

She told him to “break a leg” just before he went up on stage.
figurative translation of: break a leg
viel Glück|alles Gute|viel Erfolg|du schaffst das|Sie schaffen das

He hit the ground running, so his employer was really happy.
figurative translation of: hit the ground running
voller Begeisterung angehen|enthusiastisch angehen|hart und erfolgreich arbeiten

{Source property}
```

Figure 10: Prompt used for generating contrastive candidate pairs for the case of idioms. For the literal translation (foil) we prompt ChatGPT with the idiom in isolation. Conversely, in order to facilitate the ‘understanding’ of the idiom’s figurative connotation, for generating correct candidates we present it within the full sentence.

D Idioms Test Results

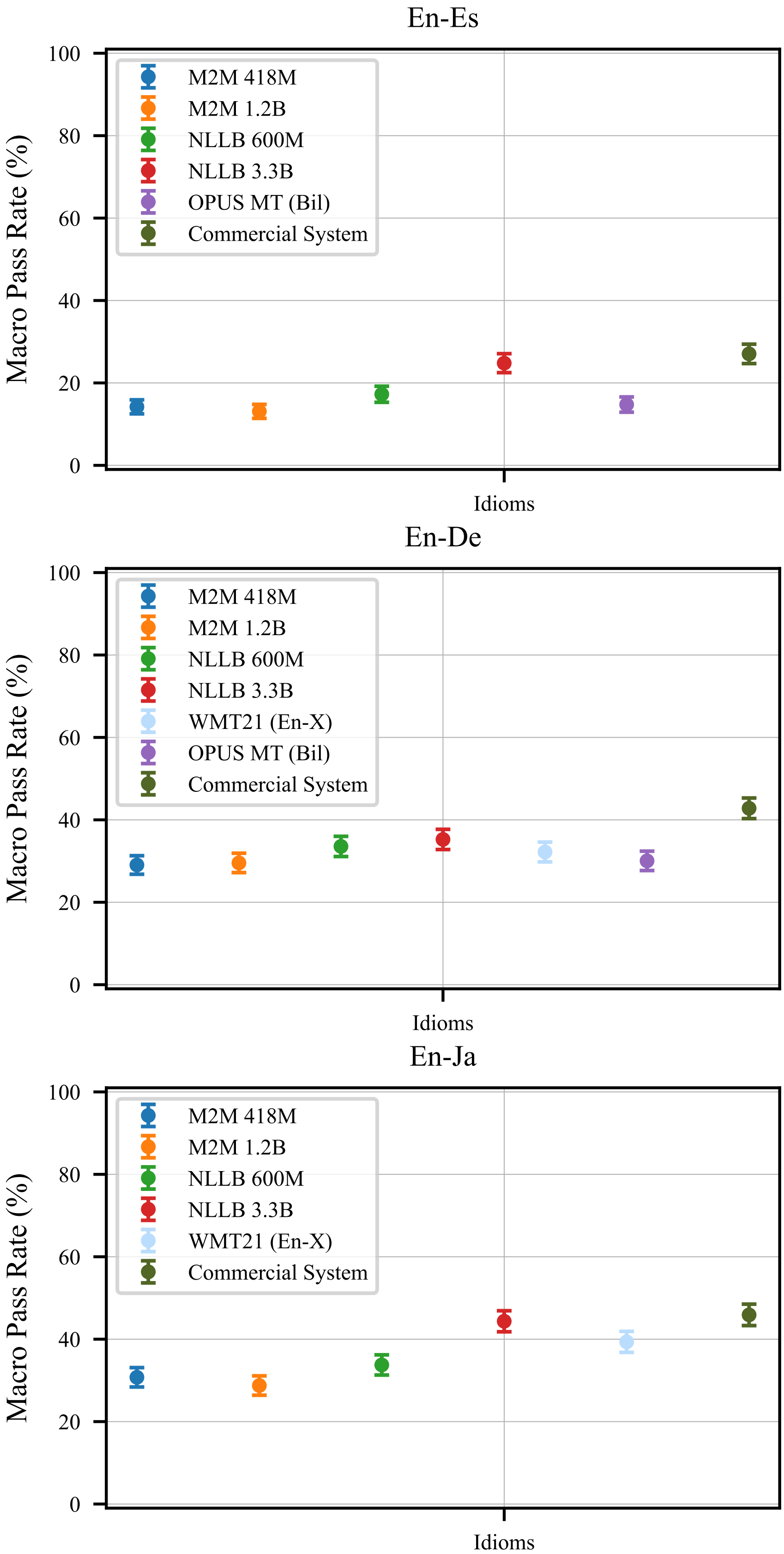

Figure 11: Idioms Test Confidence Intervals across language pairs.

E Pass Rate Confidence Intervals

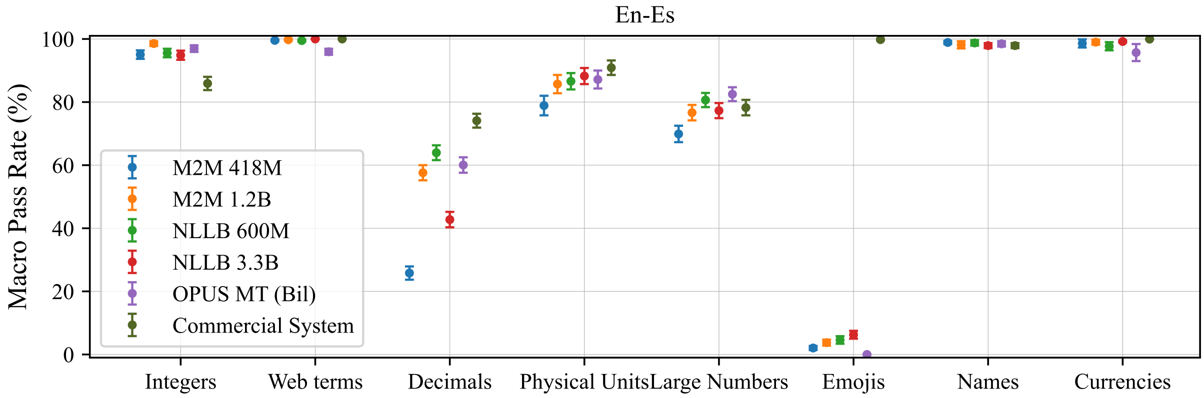

Figure 12: Macro pass rate confidence intervals for En→Es tests.

Property	Macro Pass Rate (%)					
	M2M 418M	**M2M 1.2B**	**NLLB 600M**	**NLLB 3.3B**	**OPUS MT (Bil)**	**Commercial system**
Web terms	[0.993, 0.998]	[0.995, 1.0]	[0.992, 0.998]	[1.0, 1.0]	[0.95, 0.97]	[1.0, 1.0]
Decimals	[0.236, 0.278]	[0.551, 0.602]	[0.615, 0.665]	[0.401, 0.451]	[0.576, 0.625]	[0.719, 0.764]
Integers	[0.939, 0.966]	[0.979, 0.993]	[0.943, 0.969]	[0.935, 0.963]	[0.959, 0.98]	[0.838, 0.88]
Physical Units	[0.824, 0.879]	[0.905, 0.952]	[0.915, 0.96]	[0.93, 0.972]	[0.933, 0.975]	[0.952, 0.987]
Large Numbers	[0.736, 0.787]	[0.834, 0.878]	[0.824, 0.868]	[0.795, 0.842]	[0.898, 0.935]	[0.863, 0.907]
Emojis	[0.014, 0.027]	[0.027, 0.048]	[0.035, 0.058]	[0.05, 0.075]	[0.0, 0.0]	[0.996, 1.0]
Names	[0.991, 0.994]	[0.977, 0.994]	[0.986, 0.996]	[0.978, 0.993]	[0.983, 0.993]	[0.978, 0.993]
Currencies	[0.973, 0.999]	[0.982, 0.997]	[0.985, 0.992]	[0.992, 0.998]	[0.93, 0.985]	[0.999, 1.0]
Idioms	[0.125, 0.159]	[0.114, 0.148]	[0.153, 0.192]	[0.225, 0.271]	[0.129, 0.166]	[0.247, 0.294]

Figure 13: Macro pass rate confidence intervals for En→Es tests.

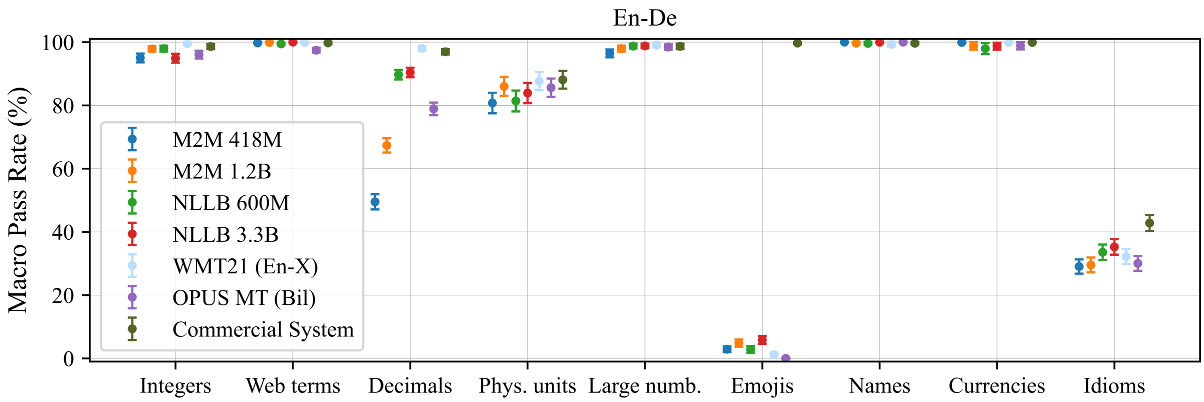

Figure 14: Macro pass rate confidence intervals for En→De tests.

Property	Macro Pass Rate (%)						
	M2M 418M	**M2M 1.2B**	**NLLB 600M**	**NLLB 3.3B**	**WMT21 (En-X)**	**OPUS MT (Bil)**	**Commercial system**
Web terms	[0.995, 1.0]	[0.998, 1.0]	[0.991, 0.998]	[1.0, 1.0]	[0.998, 1.0]	[0.967, 0.982]	[0.995, 1.0]
Decimals	[0.471, 0.519]	[0.651, 0.696]	[0.882, 0.912]	[0.889, 0.919]	[0.973, 0.987]	[0.769, 0.809]	[0.961, 0.978]
Integers	[0.936, 0.964]	[0.97, 0.987]	[0.97, 0.989]	[0.935, 0.963]	[0.99, 1.0]	[0.948, 0.973]	[0.978, 0.994]
Physical Units	[0.775, 0.84]	[0.83, 0.89]	[0.781, 0.847]	[0.807, 0.871]	[0.848, 0.905]	[0.827, 0.885]	[0.853, 0.909]
Large Numbers	[0.952, 0.977]	[0.97, 0.989]	[0.98, 0.995]	[0.981, 0.995]	[0.984, 0.997]	[0.976, 0.993]	[0.978, 0.995]
Emojis	[0.02, 0.038]	[0.037, 0.06]	[0.018, 0.039]	[0.046, 0.071]	[0.006, 0.018]	[0.0, 0.0]	[0.994, 1.0]
Names	[1.0, 1.0]	[0.993, 1.0]	[0.992, 1.0]	[0.999, 1.0]	[0.986, 1.0]	[1.0, 1.0]	[0.993, 1.0]
Currencies	[0.998, 1.0]	[0.976, 1.0]	[0.962, 0.997]	[0.976, 0.998]	[0.999, 1.0]	[0.977, 1.0]	[0.998, 1.0]
Idioms	[0.268, 0.313]	[0.272, 0.319]	[0.311, 0.36]	[0.328, 0.377]	[0.298, 0.346]	[0.277, 0.324]	[0.403, 0.453]

Figure 15: Macro pass rate confidence intervals for En→De tests.

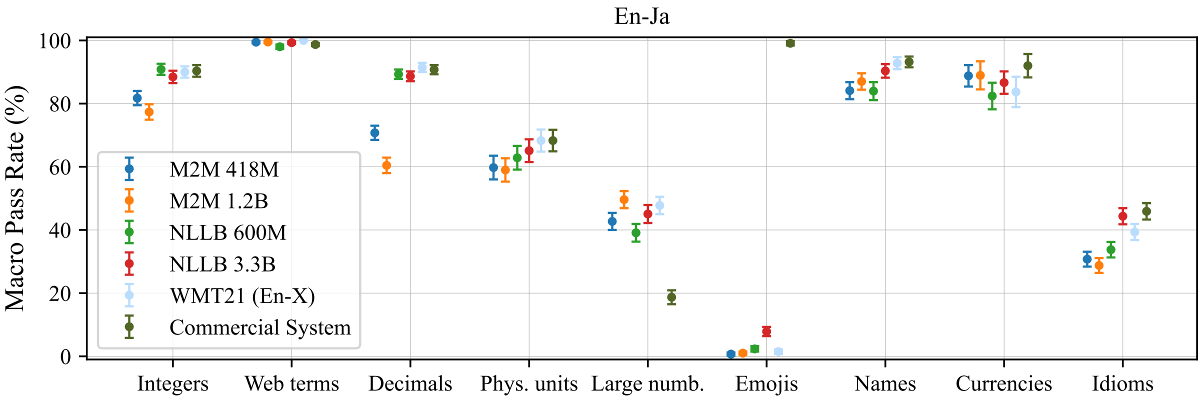

Figure 16: Macro pass rate confidence intervals for En→Ja tests.

Property	Macro Pass Rate (%)					
	M2M 418M	**M2M 1.2B**	**NLLB 600M**	**NLLB 3.3B**	**WMT21 (En-X)**	**Commercial system**
Web terms	[0.991, 0.998]	[0.992, 0.998]	[0.973, 0.987]	[0.989, 0.997]	[1.0, 1.0]	[0.982, 0.992]
Decimals	[0.685, 0.73]	[0.58, 0.629]	[0.878, 0.908]	[0.871, 0.902]	[0.9, 0.929]	[0.893, 0.922]
Integers	[0.795, 0.84]	[0.749, 0.798]	[0.891, 0.926]	[0.865, 0.904]	[0.882, 0.918]	[0.885, 0.922]
Physical Units	[0.56, 0.635]	[0.553, 0.627]	[0.591, 0.666]	[0.615, 0.687]	[0.648, 0.718]	[0.649, 0.717]
Large Numbers	[0.4, 0.454]	[0.469, 0.523]	[0.363, 0.419]	[0.422, 0.479]	[0.45, 0.505]	[0.165, 0.209]
Emojis	[0.002, 0.012]	[0.005, 0.015]	[0.015, 0.032]	[0.064, 0.093]	[0.008, 0.022]	[0.984, 0.998]
Names	[0.814, 0.868]	[0.844, 0.896]	[0.811, 0.868]	[0.882, 0.925]	[0.909, 0.947]	[0.915, 0.949]
Currencies	[0.854, 0.922]	[0.845, 0.934]	[0.782, 0.866]	[0.831, 0.902]	[0.789, 0.885]	[0.883, 0.957]
Idioms	[0.284, 0.331]	[0.264, 0.311]	[0.313, 0.362]	[0.418, 0.469]	[0.368, 0.419]	[0.433, 0.485]

Figure 17: Macro Pass Rate confidence intervals for En→Ja tests.