SALTED: A Framework for SAlient Long-tail Translation Error Detection

Vikas Raunak **Matt Post** **Arul Menezes**

Microsoft Azure AI
Redmond, Washington
{viraunak,mattpost,arulm}@microsoft.com

Abstract

Traditional machine translation (MT) metrics provide an average measure of translation quality that is insensitive to the long tail of behavioral problems. Examples include translation of numbers, physical units, dropped content and hallucinations. These errors, which occur rarely and unpredictably in Neural Machine Translation (NMT), greatly undermine the reliability of state-of-the-art MT systems. Consequently, it is important to have visibility into these problems during model development. Towards this end, we introduce SALTED, a specifications-based framework for behavioral testing of NMT models. At the core of our approach is the use of high-precision detectors that flag errors (or alternatively, verify output correctness) between a source sentence and a system output. These detectors provide fine-grained measurements of long-tail errors, providing a trustworthy view of problems that were previously invisible. We demonstrate that such detectors could be used not just to identify salient long-tail errors in MT systems, but also for higher-recall filtering of the training data, fixing targeted errors with model fine-tuning in NMT and generating novel data for metamorphic testing to elicit further bugs in models.

1 Introduction

The development of Machine Translation (MT) systems is typically guided by performance metrics (Papineni et al., 2002; Rei et al., 2020) computed on small, curated test sets, wherein system quality is often reduced to a single number. While metrics are useful for characterizing the average performance of a system, they do not provide fine-grained visibility into rarer error categories. As a result, researchers do not have a reliable way to gauge whether and to what extent a system may exhibit a wide range of negative behaviors, such as hallucinations, dropped content, or the sporadic mistranslation of important information such as names or physical units. However, such salient 'long-tail' errors undermine the reliability of MT systems, and are increasingly important to curtail in an era where MT output is often indistinguishable from that of humans (Martindale et al., 2021).

A second related problem is that many of these behaviors are rare enough that they will not be observed on standard test sets (which typically only number a few thousand sentence-pairs), even if reliable detection via references is feasible. Owing to this rarity, the detection methods that require source-reference pairs are not useful, besides being non-scalable. Consequently, a real missing feature in machine translation evaluation is having fine-grained analysis and measurements that could scale to large, unannotated datasets. In this work, we propose SALTED as a framework to tackle these challenges. Our main contributions are as follows:

1. We explore Behavioral Testing (Beizer, 1995) as a means to provide fine-grained measurements of salient long-tailed errors in MT, while addressing the challenges of rarity and scalability in obtaining those measurements.

2. We propose an iterative, specifications-based process for obtaining reliable measurements through high-precision detectors and demonstrate their utility on seven MT error classes across both research and commercial systems.

3. We demonstrate that detectors are amenable to multiple applications in MT, including higher-recall training data filtering, system-comparisons, metamorphic testing and fixing errors through fine-tuning on synthetic data.

2 The SALTED Approach

Behavioral Testing (Beizer, 1995) concerns itself with testing the input-output behavior of systems, without leveraging any knowledge about the system's internal structure. For behavioral testing

Findings of the Association for Computational Linguistics: EMNLP 2022, pages 5163–5179
December 7-11, 2022

Property	Correct Behavior Specification for Translation	Violation Example
Physical units	The model should translate the exact unit in the target language (abbreviations are allowed).	yards → Meters
Currencies	The model should translate the exact currency in the target language (both symbol abbreviations and expansions are allowed).	USD → €
Large Numbers	Large Numbers in text form should appear in the same denominations in the output.	trillions → millions
Web Terms	URLs and Web Addresses should be copied as is from the source to target, without any translation.	www.bbc.en → www.bbc.de
Numerical values	The number in a numerical value should not change beyond an allowed set of transformations (e.g., time format change, change of separators, decimal point change, number system change etc.).	24.70 → 2,470
Coverage	The model should translate the entire semantic content of the source sentence.	My friend Bob → Mi amigo
Hallucinations	The model should not produce translated content that is not grounded in the source sentence.	Hello → Hola ha ha ha ha

Table 1: **Behavior specification**: The first step in building a detector is to specify the correct expected behavior.

of Natural Language Processing (NLP) models, CHECKLIST (Ribeiro et al., 2020) proposes a process to construct test cases for evaluating different linguistic capabilities, each test case being a specified input with the associated ground truth label(s). This approach does not generalize to NMT for several reasons; (a) there could be multiple valid translations of the same input (b) errors are highly contextual: the same content may be translated accurately in one sentence, but inaccurately in another (c) errors are unstable: different model iterations may manifest errors in different sentences (d) errors are rare: a particular mistranslation may occur only once in a million sentences. Therefore, an annotated set of test cases is not just challenging to construct, but also instantly obsolete.

In SALTED, we propose a different approach to behavioral testing for NMT. Instead of relying on test cases, we translate millions of sentences and apply *detectors*. Each detector is written to detect a specific class of error, based on a specification of correct behavior. More concretely, *a detector is an algorithm, which given an input-output sentence pair returns a boolean value indicating the presence of an error condition with very* ***high precision***. The proposed detectors lie at the extreme non-trivial end of the precision-recall curve and emphasize very high precision in order to make the ensuing measurements trustworthy. At a high-level, we construct detectors by iteratively narrowing the error specifications until very high precision is achieved on a large development set. While this emphasis on precision means that a number of *potentially* erroneous output instances would not be flagged, we gain the advantage that the resulting detector could now act as a trustworthy measurement of a specific error category, a key property of useful measurements (Hand, 2016). Further, for system developers, such measurements now make previously invisible problems visible with high reliability, allowing targeted model iterations for reducing specific error types during model development.

3 Behavior Specification for MT

If we cannot specify the correct behavior, we cannot verify output correctness (Hierons et al., 2009). Therefore, the first step of constructing a detector is to specify the desired behavior that the model must satisfy with respect to the translation of certain salient content or property. Table 1 lists the behavior specifications for all the detectors we implement. We determined the 7 error classes in Table 1 owing to their disproportionate impact on user trust, since each of these specifications if violated could lead to serious consequences for user consumption of translations[1]. The desired behavior for some cases (e.g., URL translation) is unambiguous. However, in the case of expressions such

[1]Link of an example from Chemical Industry.

as physical units or currencies, an unambiguous specification is not readily apparent, e.g., NMT models are quite capable of learning to translate '10 miles' to '16 km' from parallel data. While this may be desirable for localization purposes, and successful in common cases, it could lead to dangerous inaccuracies in rarer cases, since NMT models shouldn't be trusted to do mathematics consistently and correctly, even if such conversion rules do not vary across time. Therefore, in general, our behaviour specifications require that such expressions be left semantically unchanged. This however applies only to the preservation of *semantic* content. As evident from Table 1, the specification allows changes to digit separators and to number systems (such as those between English and Chinese) in the case of numerical values. Further, as we show in section 4 this step of behavior specification is itself subject to iteration within the process of building detectors. However, an explicit enumeration of the boundary between desired and undesired behavior across different salient content types or properties provides us a useful starting point for a detector implementation.

4 Designing Detector Algorithms

Building detectors from specifications is an iterative process. A first implementation for most content types inevitably yields large numbers of false positives. It is here that our focus on *precision* over recall serves as a useful guide. In addition to producing detectors whose results are more trustworthy, this principle simplifies the task, since we can narrow in on a subset of settings that could be detected with a high certainty of correctness. Here we illustrate this process with an example.

4.1 Example: Physical units translation

Here, we consider the identification of errors in the translation of physical units, such as meters, feet, etc. for English → German translations. We decompose the process of constructing detectors into **three steps**: behavior specification, resource construction and checking for specified behavior. Each of these steps is iterated upon by quantifying the precision of error detection through human evaluation. The development iterations are done on a large initial set of monolingual source sentences and their translations generated by a MT system and the development is halted when absolute precision is achieved on this corpus. Finally, a *'test'* phase human evaluation is conducted by *varying both* the monolingual data as well as the MT system, to ascertain the final precision of the developed detectors.

Behavior Specification In this case, we start with the specification in Table 1, i.e., the desired behavior is that the physical unit measurement in the source be 'carried through' without changes in the target language. For example, '10 feet' getting translated to '10 meters' or '10 miles' getting translated to '16 km' are both errors.

Token Transformation Table Entry	Type
meter → meter, m	dist
mm → Millimeter, Millimetern, mm	dist
feet → Fuß, Füße, Fußende	dist
mile → meile, meilen	dist
km² → km², Quadratkilometer	area
sq.ft. → sq.ft., Quadratfuß, Quadratfuße	area

Table 2: A partial view of the **Token Transformation Table** constructed for use in physical unit detector. Each row comprises of allowed token transformations, along with a token 'type' annotation (used in section 7).

Transformation Table Once the desired behavior for the detector has been specified, the next step is to build the relevant resources in order to facilitate checking for the desired behavior on an arbitrary sentence pair. Table 2 illustrates the resource constructed in this case: a 'Transformation Table' of relevant source tokens which maps a source token to its set of potential translations (transformations). As we will demonstrate later, building this 'Transformation Table' is quite tractable for expressing tests that require checking for token-level transformations in translations. Further, during the process of construction of the 'Transformation Table', we also *annotate the type of the source token* (we explain its utility in section 7).

Checking for Specified Behavior Once the 'Transformation Table' has been constructed, the detector checks for the desired behavior as follows: if a source token in the transformation table is found in an input sequence, then the output sequence must contain one of the possible mappings of the source token. We also enforce that the source token must be delimited by space and that the potential target tokens need not be delimited by space in the output sequence. Note that by selectively relaxing the check on the target side,

Iteration	Algorithmic Changes	Precision
1	N/A, Initial Conditions	72.0
2	Numeric Measurements Only	94.0
3	Fixes in Transformation Table	100.0

Table 3: **Iteration vs Precision** on Physical Units Detector, measured using Human Evaluation on 100 cases flagged by error by the detector

we protect against lowering detector precision due to non-semantic/formatting changes i.e., we allow transformations such as '10 km' → '10km'.

Iterations vs Precision At each iteration, we apply the resulting detector on translations of a 1M random sample of the WMT20 English monolingual data (dataset/system details are presented in section 6.1) and measure precision by conducting a **human evaluation of 100 flagged cases**, selected at random at each iteration. Table 3 shows the resulting precision of the physical units detector across 3 iterations. At iteration 1, we observed a number of false positives pertaining to idiomatic expressions ("missed by a mile") and approximations ("a few yards further"), which were adequately translated despite missing the exact unit translation. Therefore, going from iteration 1 to 2, we narrowed our error detection only to the cases when a physical unit was preceded by a number (either in text or numeric form). This helped us avoid false positives due to the alternate senses and idiomatic uses of certain units such as feet, leading to significantly higher precision in iteration 2. Going from iteration 2 to 3, guided again by the goal of improving precision, we added/fixed entries in the transformation table to avoid false positives. The false positives obtained during each iteration of the detector are presented in appendix A.1. **We halted the manual iterations upon achieving 100% precision in human evaluation.** Table 4 presents an example of physical unit error flagged using the resulting detector. Further, we present the results of the final *'test'* evaluation in appendix B.3.

4.2 Full Suite of Detectors

The space of detector algorithms is not at all constrained by how they function as long as the contract of high-precision is satisfied. However, in this work, we mainly consider two kinds of detectors, namely token-level and sequence-level detectors.

Token-level Detectors Token-level detectors represent a generalization of the detector instance described in Section 4. Token-level detectors rely on language-pair specific transformation tables and as such, are well suited for testing the transformations of source tokens pertaining to a number of salient content types. Following the same methodology for constructing the **Physical Units** detector from Section 4, we construct detectors for evaluating the translation of salient tokens corresponding to three more content types in Table 1, namely **Currencies** (e.g., USD, $), **Large Numbers** (e.g., millions, billions) and **Web Terms** (e.g., URLs and web address terms such as https, www). Additional implementation-level details regarding the token-level detectors are provided in appendix B.1.

We also construct a token-level detector to test the translation of **numerical values**. Here, instead of a fixed transformation table, the transformation table is generated on the fly per instance. For this numerical values detector, we extract contiguous numerical values (digits) from the input sequence, condense the value's representation into a single token (by removing separators) and allow for a range of possible transformations of the numerical value, which are then checked against the output. The primary transformations considered are time conversions and date conversions. The inherent logic in this case is the same as for previous token-level detectors, except that instead of a transformation table, we construct transformation functions which are applied on the fly to generate the table. This approach of behavioral testing the translation of numerical values is quite general, unlike the explicit construction of test cases in Wang et al. (2021).

Sequence-level Detectors Sequence-level detectors do not rely on explicitly constructing language-pair specific transformation tables/functions and instead leverage more general mechanisms or resources. Such detectors are best suited for properties wherein the correct behavior can be verified using the artifacts computed from the input and the output sequences. We construct detectors for two sequence-level properties/phenomena: namely, coverage and hallucinations.

For building the **coverage detector**, we measure the number of content words (non-stopwords, non-punctuations) left unaligned using AwesomeAligner (Dou and Neubig, 2021), and label an instance as an error if the number of unaligned content words exceeds a threshold.

For constructing the **hallucination detector**, we start from the quantitative definition of hallucina-

Detector	Source-Translation Instance
Physical Unit	Teacher's hallway song and dance reminds students to stay 6 feet apart. Lehrer Flur Lied und Tanz erinnert die Schüler zu bleiben 6 Meter auseinander.
Currency	Floorpops Medina Self Adhesive Floor Tiles, £14 from Dunelm - buy now Floorpops Medina selbstklebende Bodenfliesen, 15 € von Dunelm günstig kaufen
Numerical Value	Kerridge has been an outspoken defender of his industry throughout 2020, but it was an angry Instagram post that may have made the most difference. Kerridge war das ganze Jahr über ein ausgesprochener Verteidiger seiner Branche, aber es war ein wütender Instagram-Post, der möglicherweise den größten Unterschied gemacht hat.
Coverage	Ben Cooper QC suggested it was unfair that the conspiracy theorist was arrested on May 30 while no arrests were made for breaches of lockdown restrictions at a Black Lives Matter protest taking place on the same day. Ben Cooper QC hielt es für unfair, dass der Verschwörungstheoretiker am 30.
Hallucination	The Cougars are supposed to play No. == Weblinks ==== Einzelnachweise ==

Table 4: **Detector Output examples** from the 100K WMT20 Monolingual-Evaluation set: All rows show errors made by commercial systems, as flagged by various detectors. The last row shows an error by the Microsoft system, rest show errors made by the Google system. All public APIs were accessed on January 10, 2021.

tions from Raunak et al. (2021) and adjust the thresholds for target-repeat and oscillatory hallucination detectors until high precision is achieved.

Appendix B.3 provides the results of the final *'test'* evaluation for each of the detectors, while Appendix B provides additional detector details.

5 Evaluations using SALTED

Having constructed seven high-precision detectors, we now wish to apply them to commercial systems to investigate whether we can discover any problems. We take a sample of 100K sentences from a larger 1M monolingual corpus (detailed in Section 6.1) and translate them with Google, Microsoft, and Amazon's systems by way of their paid public APIs. Table 5 shows the raw counts of erroneous translations while Table 4 presents some instances of the flagged errors from different detectors.

These results demonstrate that long-tailed errors are quite pervasive across NMT systems, despite being very rare (only **0.3% incidence rate** for Google, based on Table 11). To further validate this inexpensively with more data, we translate the full 1M monolingual corpus using the WMT21 News translation task winning system, the results (both raw counts and examples) of which are presented in Appendix D.

Property	GOOG	MSFT	AMZN
Coverage	165	1	8
Hallucinations	0	5	0
Physical Units	46	6	15
Currencies	4	1	0
Large Numbers	7	1	4
Web Content	0	0	0
Numerical Values	96	11	27
Total Errors	318	25	54

Table 5: Counts of **Erroneous Translations** found by Detectors in the 100K WMT20 Monolingual Eval Set.

6 System Comparisons & Data Filtering

A general trend in NMT is the susceptibility of trained systems to even small amounts of noisy data (Ott et al., 2018). We investigate whether detectors–optimized for error precision, rather than recall–can work as effective filters to improve systems.

6.1 Datasets and Systems

Training and Evaluation Datasets We conduct experiments on the WMT20 News Translation (English-German) task benchmark (Barrault et al., 2020). The standard WMT20 test set is used for measuring general translation performance. For behavioral testing at scale using detectors, we create a Monolingual-Evaluation set of 1M English sentences randomly sampled from the WMT20 mono-

lingual data. Due to cost constraints (e.g., in evaluating public NMT systems), we also sampled a smaller 100K Monolingual-Evaluation set. Model and training details are presented in Appendix E.

Systems We trained three systems (Table 6) each with a different training data-filtering algorithm:

- **Unfiltered (UN-F)**: The full English–German parallel training dataset provided by the WMT20 benchmark is used for training.
- **Standard (STD-F)**: We replicate the bitext filtering pipeline of Wu et al. (2020), one of the top WMT20 systems. Here, sentence-pair filtering based on maximum allowable sentence-length ratio (1:1.3) and reverse sentence-length ratio (1.3:1) is applied on the unfiltered corpus, alongside filtering sentences greater than a maximum word length (150). A language-id filter (Joulin et al., 2017) is also used, which checks if the source and target sentences are in the correct languages.
- **Detector-Based (DB-F)**: For this system, filtering as per Wu et al. (2020) is replaced by filtering using the full suite of detectors, i.e. we remove training data pairs which are flagged as erroneous by any of the detectors described in Section 4. However, the use of language-id filter is the same as in Wu et al. (2020).

Comparing Systems The comparison of the Unfiltered (UN-F) and Standard (STD-F) systems in Table 6 shows that the unfiltered system gets higher BLEU and lower TER on the WMT20 test set, apparently indicating that filtering didn't have any benefits. However, when the full suite of detectors is run on the 1M Monolingual-Evaluation set outputs, the impact of filtering becomes apparent. The Standard system incurs significantly fewer coverage errors and hallucinations as well as fewer errors in the translation of numerical values and currencies. These measurements bring to light the previously hidden impact of filtering since the standard metrics aren't able to capture these trade-offs in model behaviors, achieving only similar scores.

Data Filtering using Detectors The third column in Table 6 shows the measurements for the system trained on data filtered using the suite of detectors. The results show that the DB-F system achieves higher BLEU than both the Standard and Unfiltered systems, while yielding similar results

Measurement	UN-F	STD-F	DB-F
Training Data	**48.2M**	36.9M	41.7M
BLEU ↑	32.4	31.4	**32.9**
ChrF2++ ↑	58.4	58.0	**58.8**
COMET ↑	42.0	38.1	**45.7**
TER ↓	54.5	55.5	**54.2**
Coverage ↓	742	**309**	365
Hallucinations ↓	37	**0**	8
Physical Units ↓	141	151	**126**
Currencies ↓	17	**7**	13
Large Numbers ↓	113	**60**	67
Web Terms ↓	43	39	**33**
Numerical Values ↓	1,000	503	**429**

Table 6: **Metric Based** System Comparisons on the WMT20 Test set and **Detector Based** system comparisons on the 1M Mono-Eval Set for the three systems. Note that ↓ implies lower is better, ↑ implies otherwise.

to STD-F on the various long tail error categories. This indicates the benefits of a more targeted approach to filtering through high-precision detectors, which helps the model strike a better trade-off between preserving general model performance and preventing long-tailed translation errors. However, it is also clear that filtering *alone* is not sufficient in reducing salient long-tail errors, e.g., the number of errors in the physical unit category is relatively unchanged. In Section 8, we show how SALTED could be used to inject correct model behavior through data synthesis.

6.2 Discussion

The results show that a very targeted multi-dimensional view of model behavior could be developed through the use of detectors, bringing visibility into fine-grained model performance issues not evident through traditional metrics. The results also show that the same detectors could be used as an alternative to standard corpus filtering (Wu et al., 2020), leading to better model performance.

7 Metamorphic Testing

The low incidence rates of long-tail errors necessitate large amounts of data in order to elicit them. The SALTED framework further addresses this problem through *metamorphic* testing, wherein new test inputs are produced by modifying an input instance in systematic ways and the outputs are then tested for correctness through detectors.

Sequence Type	Instance	Algorithm Step
Source	The plesiosaur teeth it self is about 43 **mm** long.	
Reference	Der Plesiosaurier Zahn selber misst etwa 43 **mm**.	Sentence Selection
Templatized Source	The plesiosaur teeth it self is about 43 [*VAL*] long.	
Templatized Reference	Der Plesiosaurier Zahn selber misst etwa 43 [*VAL*].	Templatization
Meta Source Instance	The plesiosaur teeth it self is about 43 ***feet*** long.	
Meta Reference Instance	Der Plesiosaurier Zahn selber misst etwa 43 ***Fuß***.	Type Substitutions

Table 7: **Meta-Corpus** instance generation example using the physical units detector in Algorithm 1.

Experiment Given a token-level detector and an initial corpus of monolingual (source) sentences, if a source token in the detector's transformation table is found in a source sentence, delimited by space on either sides, we create new instances by substituting that token with others of the same type (as annotated in Table 2). For example, a sentence with the word 'meters' can be changed to one with the word 'yards'. The new sentences can then be translated by a system, and the detectors applied to these novel (input, translation) pairs.

Property	New Sentences	Novel Cases
Physical units	204,029	4,503
Currencies	1,232,988	775
Large Numbers	7,885	42
Web Content	8,238	196

Table 8: **Metamorphic Testing**: New instances elicit novel error cases in our research system.

Results and Discussion Results are presented in Table 8. We find that SALTED metamorphic testing elicits a number of novel bugs, providing new data points/instances for investigating system errors or comparing system performance. Unlike traditional MT metamorphic tests (e.g., Gupta et al. (2020)), the metamorphic testing enabled by the SALTED framework leverages detectors for error checks and is therefore high-precision by default.

8 Fixing Salient Long-Tailed Errors

While data filtering can improve models by removing erroneous data, it doesn't guarantee that there are sufficient correct examples of a *type* to learn correct behavior from. In this section, we leverage detectors to generate an example-dense synthetic corpora for fixing model errors via finetuning.

Meta-Corpus Algorithm 1 describes the generation of a synthetic corpus wherein we leverage the detectors to ensure that the generated sentence pairs are correct with respect to a particular measurement. An example illustrating the steps is presented in Table 7. The algorithm consists of three steps: a sentence-pair selection step where a sentence is selected for templatization if the detector does not deem it erroneous, a templatization step for the tokens in the transformation table within the selected sentence pair and finally generation of new sentence pairs by substituting the templatized tokens with (source, target) tokens of the same type.

Algorithm 1: Meta-Corpus Generator

Data: Parallel Dataset S of size n, token-level Detector A
Result: Meta-Corpus M, Templates T of Size k
for *i = 1 to n* **do**
  `/* Sentence Pair Selection */`
  Apply Detector A on S_i;
  If S_i has errors: continue;
  `/* Templatization */`
  Else: Templatize S_i and add to T
for *i=1 to k* **do**
  `/* Type Substitutions */`
  Substitute T_k with Source-Target Token Mappings of the same Type
  Store the Generated Sentence Pairs in M

Experiment We generate a 'meta-corpus' (Algorithm 1) using the physical units detector on a random sample of the WMT20 training data of size 1M. We then finetune, for 3 epochs, the best checkpoint of the Standard model using a 1:1 mixture of the sentence pairs sampled from the Meta-Corpus and the general 1M training data, filtered using the same detector. We measure general performance on the WMT20 test set and targeted performance on the translation of physical units on the 100K Mono-Evaluation set. Finetuning learning rates are provided in Appendix E.6.

Results and Discussion The results (Table 10) show that just 10K or 20K 'correct' examples, provided by the Meta-Corpus are sufficient to reduce the number of physical unit errors flagged by the detector, while preserving the general model perfor-

Sequence Type	Instance
Source	Remind kids to keep their masks up and stay at least six feet apart.
Baseline Output	Erinnern Sie Kinder, um ihre Masken zu halten und bleiben mindestens sechs Meter auseinander.
Finetuned Output	Erinnern Sie Kinder, um ihre Masken zu halten und bleiben mindestens sechs Fuß auseinander.

Table 9: **Meta-Corpus Based Finetuning**: Accompanying Table 7, this example shows an case where a token-level error ('feet' → 'Meter') was fixed ('feet' → 'Fuß') by applying finetuning using the synthetic Meta-Corpus.

Model	MC Size	Error Cases	BLEU
Baseline	None	19	31.4
Finetuned	10K	4	31.6
Finetuned	20K	2	31.4
Finetuned	50K	3	31.5

Table 10: **Finetuning** on Meta-Corpus leads to reduction in physical units errors. *MC Size* is the size of the Meta-Corpus; BLEU scores are reported on WMT20.

mance. An example of this error fix is presented in Table 9. A limitation of this approach is that it can only fix mistranslations, not dropped content. In fact, the few cases that remain in the case of Finetuned model (20K) are the cases where the unit is dropped along with a clause in the source sentence.

9 Related Work

Quality Estimation for MT The task of Quality Estimation (QE) is concerned with determining the quality of a translation without access to any reference (Specia et al., 2018, 2020). In particular, sentence-level QE allows the development of models which act as metrics in the absence of references (QE-as-a-metric). However, such QE-as-a-metric models still focus on a combined evaluation of adequacy and fluency, rendering them insensitive to the presence of long-tailed errors. E.g., consider the two translations in Table **??**. The state-of-the-art COMET (Rei et al., 2020) QE-as-a-metric model (detailed in Appendix E.7) produces a score of **8.73** for the **baseline** output and **6.00** for the **finetuned** output, even though the latter is clearly the correct translation. In Appendix F, we present further quantitative experiments to illustrate this. Note that SALTED correctly flags these errors.

Further, we claim that this insensitivity to a long-tailed errors is not due to deficient modeling of the particular neural QE model, but due to a fundamental limitation of leveraging neural models such COMET (Rei et al., 2020) for evaluation as well as error detection (Sudoh et al., 2021). Even though the recent trend in the NLP community has been towards learning neural metrics, we argue that this paradigm isn't equipped to tackle the problem of salient long-tail evaluation since robust interpolation in neural networks requires many orders of magnitude higher number of parameters than currently employed (Bubeck and Sellke, 2021), which implies that evaluation using neural models is likely to remain suspect at the long-tail.

Behavioral Testing for MT A number of previous works (He et al., 2020; Gupta et al., 2020; Sun et al., 2020; Wang et al., 2021; He et al., 2021) have tried to construct tests for eliciting errors in NMT systems' behavior. We present a comparison of SALTED against these works in appendix G.

10 Conclusions

In this paper, we have advocated for and demonstrated the utility of a principled, specifications-based approach to reliably flag salient long-tailed MT errors through high-precision detectors. We introduced an iterative, precision-driven process for developing such detectors and applied it on seven classes of MT errors, eliciting a range of errors from state-of-the-art research and commercial systems. Although the manual development of such detectors incurs significant cost, the resulting payoff is high with the constructed detectors applicable universally across different systems and datasets. Further, we demonstrated the utility of SALTED for four different use cases in MT: for obtaining reliable measurements of salient long-tailed errors in translations of arbitrary monolingual data, for corpus filtering, for system comparisons, and for fixing token-level errors through a synthetically generated meta-corpus that teaches the model to learn correct behaviors. We hope that our work serves as a useful step towards more reliable MT.

11 Limitations

While our work highlighted a number of long-tailed errors in state-of-the-art translation MT systems, a limitation of the approach is the effort required in manually defining specifications and then constructing detectors based on the defined specifications. Our proposed approach might also face hurdles in adoption since, to the best of our knowledge, the idea of specifications without any references is novel in large-scale MT evaluation, and has been mainly developed and used in software/hardware engineering research. Another limitation of the work is that fixing salient long-tail errors is only feasible when the errors are mistranslations, rather than dropped content (e.g., dropped clause or phrase). If the salient errors are generated by long chunks of source sentences remaining untranslated by the model, then the proposed fine-tuning approach will not improve such translations.

References

Loïc Barrault, Magdalena Biesialska, Ondřej Bojar, Marta R. Costa-jussà, Christian Federmann, Yvette Graham, Roman Grundkiewicz, Barry Haddow, Matthias Huck, Eric Joanis, Tom Kocmi, Philipp Koehn, Chi-kiu Lo, Nikola Ljubešić, Christof Monz, Makoto Morishita, Masaaki Nagata, Toshiaki Nakazawa, Santanu Pal, Matt Post, and Marcos Zampieri. 2020. Findings of the 2020 conference on machine translation (WMT20). In *Proceedings of the Fifth Conference on Machine Translation*, pages 1–55, Online. Association for Computational Linguistics.

Boris Beizer. 1995. *Black-box testing: techniques for functional testing of software and systems*. John Wiley & Sons, Inc.

Sebastien Bubeck and Mark Sellke. 2021. A universal law of robustness via isoperimetry. In *Advances in Neural Information Processing Systems*.

Alexis Conneau, Kartikay Khandelwal, Naman Goyal, Vishrav Chaudhary, Guillaume Wenzek, Francisco Guzmán, Edouard Grave, Myle Ott, Luke Zettlemoyer, and Veselin Stoyanov. 2020. Unsupervised cross-lingual representation learning at scale. In *Proceedings of the 58th Annual Meeting of the Association for Computational Linguistics*, pages 8440–8451, Online. Association for Computational Linguistics.

Zi-Yi Dou and Graham Neubig. 2021. Word alignment by fine-tuning embeddings on parallel corpora. In *Proceedings of the 16th Conference of the European Chapter of the Association for Computational Linguistics: Main Volume*, pages 2112–2128, Online. Association for Computational Linguistics.

Shashij Gupta, Pinjia He, Clara Meister, and Zhendong Su. 2020. Machine translation testing via pathological invariance. In *Proceedings of the 28th ACM Joint Meeting on European Software Engineering Conference and Symposium on the Foundations of Software Engineering*, ESEC/FSE 2020, page 863–875, New York, NY, USA. Association for Computing Machinery.

David J Hand. 2016. *Measurement: A very short introduction*. Oxford University Press.

Pinjia He, Clara Meister, and Zhendong Su. 2020. Structure-invariant testing for machine translation. In *Proceedings of the ACM/IEEE 42nd International Conference on Software Engineering*, ICSE '20, page 961–973, New York, NY, USA. Association for Computing Machinery.

Pinjia He, Clara Meister, and Zhendong Su. 2021. Testing machine translation via referential transparency. In *2021 IEEE/ACM 43rd International Conference on Software Engineering (ICSE)*, pages 410–422. IEEE.

Robert M. Hierons, Kirill Bogdanov, Jonathan P. Bowen, Rance Cleaveland, John Derrick, Jeremy Dick, Marian Gheorghe, Mark Harman, Kalpesh Kapoor, Paul Krause, Gerald Lüttgen, Anthony J. H. Simons, Sergiy Vilkomir, Martin R. Woodward, and Hussein Zedan. 2009. Using formal specifications to support testing. *ACM Comput. Surv.*, 41(2).

Armand Joulin, Edouard Grave, Piotr Bojanowski, and Tomas Mikolov. 2017. Bag of tricks for efficient text classification. In *Proceedings of the 15th Conference of the European Chapter of the Association for Computational Linguistics: Volume 2, Short Papers*, pages 427–431, Valencia, Spain. Association for Computational Linguistics.

Taku Kudo and John Richardson. 2018. SentencePiece: A simple and language independent subword tokenizer and detokenizer for neural text processing. In *Proceedings of the 2018 Conference on Empirical Methods in Natural Language Processing: System Demonstrations*, pages 66–71, Brussels, Belgium. Association for Computational Linguistics.

Marianna Martindale, Kevin Duh, and Marine Carpuat. 2021. Machine translation believability. In *Proceedings of the First Workshop on Bridging Human–Computer Interaction and Natural Language Processing*, pages 88–95, Online. Association for Computational Linguistics.

Myle Ott, Michael Auli, David Grangier, and Marc'Aurelio Ranzato. 2018. Analyzing uncertainty in neural machine translation. *CoRR*, abs/1803.00047.

Myle Ott, Sergey Edunov, Alexei Baevski, Angela Fan, Sam Gross, Nathan Ng, David Grangier, and Michael Auli. 2019. fairseq: A Fast, Extensible Toolkit for Sequence Modeling. In *Proceedings of the 2019 Conference of the North American Chapter of the*

Association for Computational Linguistics (Demonstrations), pages 48–53, Minneapolis, Minnesota. Association for Computational Linguistics.

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.

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, Arul Menezes, and Marcin Junczys-Dowmunt. 2021. The curious case of hallucinations in neural machine translation. In *Proceedings of the 2021 Conference of the North American Chapter of the Association for Computational Linguistics: Human Language Technologies*, pages 1172–1183, Online. 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.

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.

Lucia Specia, Frédéric Blain, Marina Fomicheva, Erick Fonseca, Vishrav Chaudhary, Francisco Guzmán, and André F. T. Martins. 2020. Findings of the WMT 2020 shared task on quality estimation. In *Proceedings of the Fifth Conference on Machine Translation*, pages 743–764, Online. Association for Computational Linguistics.

Lucia Specia, Carolina Scarton, and Gustavo Henrique Paetzold. 2018. Quality estimation for machine translation. *Synthesis Lectures on Human Language Technologies*, 11(1):1–162.

Katsuhito Sudoh, Kosuke Takahashi, and Satoshi Nakamura. 2021. Is this translation error critical?: Classification-based human and automatic machine translation evaluation focusing on critical errors. In *Proceedings of the Workshop on Human Evaluation of NLP Systems (HumEval)*, pages 46–55, Online. Association for Computational Linguistics.

Zeyu Sun, Jie M. Zhang, Mark Harman, Mike Papadakis, and Lu Zhang. 2020. Automatic testing and improvement of machine translation. In *Proceedings of the ACM/IEEE 42nd International Conference on Software Engineering*, ICSE '20, page 974–985, New York, NY, USA. Association for Computing Machinery.

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.

Shuangzhi Wu, Xing Wang, Longyue Wang, Fangxu Liu, Jun Xie, Zhaopeng Tu, Shuming Shi, and Mu Li. 2020. Tencent neural machine translation systems for the WMT20 news translation task. In *Proceedings of the Fifth Conference on Machine Translation*, pages 313–319, Online. Association for Computational Linguistics.

A Designing Detector Algorithms

The development of detectors is a manual, rules-driven iterative process with the goal of constructing a very high precision error detector which could be trusted as a measurement of a specific error category. In section 4, we presented an example of this process for a token-level detector (physical units).

A.1 False Positive Examples

We provide a few examples of the false positives obtained in the first two iterations of constructing the physical units detector, the precision of which is enumerated in Table 3. Examples in Table 11 show a representative sample of the false positives at each iteration. In the first iteration, the error criteria didn't target numeric measurements only, as a result, we got false positives where the change of unit didn't imply semantic change. In the second iteration, we got errors pertaining to an incomplete transformation table, where 'Morgen' wasn't specified as a potential translation for the unit 'acres'.

B Full Suite of Detectors

We provide more details on the implementation of detectors. The process of construction of detectors remains the same as described in section 4. For each of the detectors, the iterative process is halted when the precision of the error detector reaches 100. This precision is measured using human evaluation, by randomly sampling 100 error instances obtained by applying the detector on 1M source-translation pairs. The sources for obtaining these translations are obtained by randomly sampling the WMT20 monolingual data.

Iteration	Source-Translation Instance
1	Closed-circuit cameras watch over every inch of the main street. Closed-Circuit-Kameras wachen über jeden Zentimeter der Hauptstraße.
1	The officiant of the wedding then rushed the family away from the beach, back towards a large house several yards away. Der Beamte der Hochzeit eilte dann die Familie vom Strand weg, zurück in Richtung eines großen Hauses, das mehrere Meter entfernt war.
2	The city of Anaheim tweeted around 2:30 p.m. that the fire was estimated at 700 acres . Die Stadt Anaheim twitterte gegen 14:30 Uhr, dass das Feuer auf 700 Morgen geschätzt wurde.
2	California's footprint was even larger: Fires there have now consumed about 3.1 million acres - a modern record. Kalifornien Fußabdruck war sogar noch größer: Feuer dort haben jetzt etwa 3,1 Millionen Morgen verbraucht - ein moderner Rekord.

Table 11: Examples of **False Positives** in the first two iterations of the physical units detector.

B.1 Token-Level Detectors

Token-level detectors rely on the construction of transformation tables or transformation functions, that map source tokens to their potential mappings in the target language. In the next sections, we provide implementation-level details regarding token-level detectors.

B.1.1 Physical Units Detector

For physical units detector, the entries in the transformation table contain units associated with distance (miles, meters, centimeter, millimeter, inch, kilometre, feet, yard), area (square kilometre, square metre, acres), weight (kilogram, pound), volume (litres, cubic mm) and temperature (celsius, fahrenheit). A number of derivative units follow automatically: e.g., an error translation of 'km/hr' getting translated to 'miles/hr' could also be detected using the entry for 'km' in the transformation table.

Token Transformation Table Entry	Type
dollar → dollar, usd, dollars, $	text
$ → $, dollar, dollars, usd	sym
rupees → ₹, rupie, rupien, rupee(s), rs	text
₹→ ₹, rupie, rupien, rupee, rupees, rs	sym

Table 12: A partial view of the **Token Transformation Table** constructed for use in currency detector. Each row comprises of allowed token transformations, along with a token 'type' annotation (either symbol or text in this case).

B.1.2 Currency Detector

For currency detector, a partial view of the transformation table is presented in 12. The full entries in the currency table comprise of 20 currencies. We obtained similar false positives as for the physical units detector in Appendix A until we didn't allow exceptions for idiomatic expressions (e.g., 'pennies on the dollar') or approximations (e.g., 'a few dollars').

B.1.3 Large Numbers

For the large numbers detector, we build a transformation table for the text version of larger numbers ('million(s)', 'billion(s)', 'trillion(s)'). We check for their translations into both text and numeric forms.

B.1.4 Web Terms

For the detector corresponding to web terms, we make use of both transformation table as well as a transformation function. We check for the correct translation of URLs (which is copying behavior in this case) extracted from the source as well as the correct translation (again, copying in this case) of web terms such as https, www and ftp. Therefore, for this detector, the transformation table comprises only of identity mappings and the transformation function acting on the extracted URL is also an identity mapping.

B.1.5 Numerical Values Detector

The numerical values detector allows transformations of the extracted numerical value into a range of possible translations: time-conversions (e.g., '2:00' to '14:00'), date conversions (e.g., 'mm/dd/yyyy' to 'dd/mm/yyyy'), separator changes (e.g., '10,000' in English to '10,000' in German) and numeric to text forms (e.g., '12' to 'zwölf').

B.2 Sequence-Level Detectors

Language	Coverage	Hallucinations
Russian	5	0
Dutch	6	1
Danish	1	6
Swedish	12	0
Spanish	6	0

Table 13: Number of **Erroneous Translations** flagged by the language-agnostic sequence-level detectors for translations into multiple languages.

B.2.1 Coverage

As described in section 4.2, for implementing the coverage detector, we make use of alignments obtained through a multilingual BERT-based aligner. To compute the number of unaligned tokens in the source, after computing the alignments we filter the source tokens by removing stop-words compiled from a number of sources as well as by removing punctuation tokens. The coverage detector then flags a translation if the number of unaligned content tokens exceed a threshold. This threshold is bucketized in terms of the source sentence length. We use a threshold of 10 if the input sentence length is less than 50 tokens, 20 if input sentence length is between 50 and 100, 30 if the input sentence length is between 100 and 200 and 40 otherwise.

B.2.2 Hallucinations

The hallucination detector tries to count the number of oscillatory and natural hallucinations (Raunak et al., 2021). The detection of oscillatory hallucinations is done by the following algorithm: if the count of the most frequent bigram in the output exceeds the count of the most frequent bigram in the source by 4 and the count of the most frequent output bigram exceeds 10, then it is flagged as an oscillatory hallucination. For detecting natural hallucinations, we compute the number of the unique sources getting translated to the same output, and the output is deemed as a natural hallucination if 5 or more source sentences, each with different lengths translate to it. The hallucination count is then reported by combining the number of natural and oscillatory hallucinations. Note that both the counts are computed independently of each other. For example, in section 5, we found that one of the commercial translators (Microsoft) incurs 5 natural hallucinations, without incurring any oscillatory hallucinations. For illustration, we present these hallucination cases in Table 14.

Source-Translation Instance
The Cougars are supposed to play No. == Weblinks ==== Einzelnachweise ==
Ms. Williams was only seeded No. == Weblinks ==== Einzelnachweise ==
"Geomsanaejeon" a.k.a. == Weblinks ==== Einzelnachweise ==
Greg Brown (No. == Weblinks ==== Einzelnachweise ==
Downtown L. A. == Weblinks ==== Einzelnachweise ==

Table 14: Examples of **Hallucinations** in one of the Commercial Translation Systems (Microsoft). The public API was accessed on January 10, 2021.

Detector	Error cases
Coverage	444
Hallucinations	108
Physical Units	133
Currencies	22
Large Numbers	84
Web Content	30
Numerical Values	405
Total Errors	1226

Table 15: Number of **Erroneous Translations** flagged by detectors for the WMT21 News Translation task winning system.

B.3 Test-Phase Evaluation of Detectors

To conduct a test-phase evaluation of detectors (the development iterations are halted when absolute precision is achieved on the large initial develop-

ment corpus) we vary both the monolingual data as well as the system generating the translations. We translate a separate randomly sampled 250K monolingual corpus using the WMT21 winning system and measure the precision of each of the detectors through human evaluation on the flagged input-output pairs. The precision numbers are presented below for each of the detectors. We obtain absolute precision for each of the detectors on all except one of the detectors: Numerical Values (92.53 percent, with 5 false positives). These false positives from the Numerical Values detector pertained to the handling of fractions in certain non-standard forms, which were parsed incorrectly by the detector (an example is presented in Table 17).

Detector	Error cases	Precision
Coverage	70	100.0
Hallucinations	1	100.0
Physical Units	33	100.0
Currencies	4	100.0
Large Numbers	9	100.0
Web Content	7	100.0
Numerical Values	67	92.53

Table 16: Number of **Erroneous Translations** flagged by detectors for the WMT21 News Translation task winning system on 250K 'Test' Monolingual data, alongside Precision as adjudged by human evaluation.

C Sequence-Level Detector Applications

We translate the 100K monolingual sentences into 5 different languages using a commercial system (Microsoft) and measure the number of coverage and hallucination errors. We find that the same thresholds used for English-German apply well to the languages in Table 13 too, with the flagged outputs exhibiting the related error conditions.

D Examples from WMT21 Winning System

In this section, we translate the 1M Monolingual Evaluation set using the WMT21 News Translation task winning system. Beam size of 5 was used for generating the translations [2]. We report the detector error counts in Table 15 and examples for different error categories in Table 19. Table 19 shows the error instances from different detectors. Here, counts and the examples show that a range of long-tail errors errors persist in the WMT21 system as well, with a **0.12% incidence rate**, similar to that of the commercial systems.

E Experimental Details

For experiments, we use fairseq (Ott et al., 2019). Sentencepiece (Kudo and Richardson, 2018) with a joint token vocabulary of 32K was learned over the training corpus. The Transformer model used, comprising of 6 layers with embedding size 512, FFN layer dimension 4096 and 16 attention heads, was trained for 100 epochs, with the best checkpoint selected using the loss score on the validation set. Additional experimental details are provided in appendix E. For, BLEU, TER, ChrF2++ evaluations SacreBLEU is used (Post, 2018), for COMET scores the implementation provided by Rei et al. (2020) is used. All models were trained on 8 Nvidia V100 GPUs and a beam size of 5 was used for each evaluation.

E.1 Data Sources and Filtering

Table 18 lists the data sources used for training the models in section 6. The Monolingual evaluation set was sampled from one of the WMT20 monolingual data sources [3]. Further, the language-id filter for the STD-F baseline in 6 was built using the more accurate (larger) version of the fasttext released models (Joulin et al., 2017) [4].

E.2 Transformer Training

For each of the models, a dropout of 0.1 was used (including relu-dropout and attention-dropout in (Ott et al., 2019)). The optimizer used was Adam with the adam-betas parameters set to (0.9, 0.98). Clip-norm of 1.2 was used. For each of the models the encoder-decoder embeddings were tied. Each of the models were trained using a maximum batch size of 4096 tokens. Further, 3K warmup updates were used, with the initial learning rate set to 1e-7 and the learning rate set to 1e-4. The batch size was set to 4096 tokens, and the update frequency was set to 200. In each case, the inverse-sqrt learning rate scheduler was used, along with fp16 mode training.

E.3 Sentencepiece Vocabulary

For each of the models, the 32K Unigram LM-based sentencepiece [5] vocabulary was constructed

[2] https://github.com/pytorch/fairseq/tree/main/examples/wmt21

[3] https://data.statmt.org/news-crawl/en/

[4] https://fasttext.cc/docs/en/language-identification.html

[5] https://github.com/google/sentencepiece

Sequence Type	Instance
Source	Just as he had lost the first set about 1.1/2 hours earlier but turned things around, with the help of a dip in level from the fourth-seeded Zverev.
Translation	So wie er etwa eineinhalb Stunden zuvor den ersten Satz verloren hatte, aber mit Hilfe eines Levelrückgangs des an vierter Stelle gesetzten Zverev die Wende schaffte.

Table 17: False Positive Example for the Numerical Values Detector

Data Source	Sentence Pairs
Europarl	1,828,521
ParaCrawl	34,371,306
Common Crawl	2,399,123
News Commentary	361,445
Wiki Titles	1,382,625
Tilde Rapid	1,631,639
WikiMatrix	6,227,188
Total	48,201,847

Table 18: The WMT20 **Data sources** used for training the English-German models in section 6

by using a character coverage of 0.9995, on 3M randomly sampled sentences from the training corpus.

E.4 SacreBLEU Configuration signatures

The WMT20 En-De SacreBLEU configuration signature for BLEU computation is nrefs:1|case:mixed|eff:no|tok:13a|smooth:exp|version:2.0.0, for ChrF2++ the signature is nrefs:1|case:mixed|eff:yes|nc:6|nw:2|space:no|version:2.0.0 and for TER the signature is nrefs:1|case:lc|tok:tercom|norm:no|punct:yes|asian:no|version:2.0.0.

E.5 Meta-Corpus Fine-tuning

For meta corpus generation, the substitutions are made using Algorithm 1 only on the (source, target) pairs with one occurrence of the physical unit on each side, i.e. only one occurrence of the physical unit is templatized on each side, as illustrated in table 9.

E.6 Meta-Corpus Fine-tuning

For finetuning, in each case, we use 1K warmup updates, with warmup initial learning rate set to 1e-7 and the learning rate set to 4e-4. Rest of the parameter details remain the same as in Appendix E.2.

E.7 COMET QE-as-a-metric model

The COMET QE-as-a-metric model is built on top of XLM-R (large) (Conneau et al., 2020) and is trained on Direct Assessment (DA) scores from WMT 17-19.

F Insensitivity Towards Long-Tailed Errors

We add two experiments to quantitatively substantiate the claim the state-of-the-art QE metrics cannot detect salient long-tailed errors with high precision.

The **first experiment** is as follows: We select 100 flagged error cases (from the UN-F baseline in Table 6), of the physical units detector (we further verify that these are indeed salient long-tailed errors). Note that these error cases were obtained by applying the detector on 1M translations. We then obtain the Comet-QE scores for each of those 1M translations. We sort the source-translation pairs based on the Comet-QE scores, and measure how many of the flagged cases are present in the worst-K scoring translations. We tabulate this, i.e. how many of the flagged error cases were present in the worst-K scoring translations in Table 20 below. Even in the 100K worst scoring sentences, only 3 out of 100 erroneous translations were present. This shows that the existing state-of-the-art COMET-QE model is insensitive to the salient long-tailed errors pertaining to physical units. Further, we find that this insensitivity holds true across different salient long-tailed error categories.

For the **second experiment**, instead of sorting the translations based on Comet-QE scores, we measure how many of the erroneous cases were present in the set of translations that scored below a certain threshold. Table 21 presents the results. This shows that the erroneous cases are spread across a range of scores.

Detector	Source-Translation Instance
Hallucination	Lampard and Mourinho exchanged barbs as old friends became enemiesCredit : PA : Press Association in Lampard and Mourinho exchanged barbs as old friends became enemiesCredit : PA : Press Association (PA : PA : PA : PA : PA : PA : PA) Lampard and Mourinho exchanged barbs as old friends became enemiesCredit : PA : PA : Press Association (PA : PA : PA)
Web Content	Go to the Income Tax Department website by typing https://www.incometaxindiaefiling.gov.in/home in the address bar of your browser. incometaxindiaefiling.gov.in/home in die Adressleiste Ihres Browsers.
Physical Units	Scott McLaughlin came close from a free kick 30 yards out. Scott McLaughlin kam mit einem Freistoß aus 30 Metern in die Enge.
Large Number	With support from the array of diplomatic associations, A.F.S.A. has so far raised three quarters of a million dollars, Rubin said. The funds are helping defray the legal fees of seven witnesses, covering in full charges that were not paid by the State Department or waived through pro-bono assistance. "It's very moving that current and former Foreign Service officers, most of whom don't have much money, have contributed to help their colleagues," John Bellinger, who served as a legal adviser to the State Department and National Security Council during the George W. Bush Administration, told me. He and a former C.I.A. general counsel, Jeff Smith, represented Ambassador Taylor and Ambassador Mike McKinley. But Taylor, who was not a member of the Foreign Service and was pulled out of retirement to return to Ukraine after Ambassador Marie Yovanovitch was recalled, is not a member of A.F.S.A. - and thus not eligible for its financial aid. He was in Ukraine for only six months. Volker is also not a member of A.F.S.A. And none of the witnesses from the White House, Department of Defense, or the Office of Management and Budget qualifies for its aid, either. "Es ist sehr bewegend, dass aktuelle und ehemalige Beamte des Auswärtigen Dienstes, von denen die meisten nicht viel Geld haben, dazu beigetragen haben, ihren Kollegen zu helfen", sagte mir John Bellinger, der während der Regierung von George W. Bush als Rechtsberater für das Außenministerium und den Nationalen Sicherheitsrat tätig war. Er und ein ehemaliger C.I.A. General Counsel, Jeff Smith, vertraten Botschafter Taylor und Botschafter Mike McKinley. Aber Taylor, der kein Mitglied des Auswärtigen Dienstes war und aus dem Ruhestand gezogen wurde, um in die Ukraine zurückzukehren, nachdem Botschafterin Marie Yovanovitch zurückgerufen wurde, ist kein Mitglied des A.F.S.A. - und somit nicht für seine finanzielle Unterstützung berechtigt. Er war nur sechs Monate in der Ukraine. Volker ist auch kein Mitglied der A.F.S.A. Und keiner der Zeugen aus dem A.F.S.A., dem Haushalts- und Verteidigungsministerium des Weißen Hauses, qualifiziert
Coverage	James Hill of Diamond Bay loves the directive on his local church message board : "Thou shalt wear a mask - Hygenesis 20 : 20" thou shalt wear a mask - Hygenesis 20 : 20 "(Du sollst eine Maske tragen - Hygenesis 20 : 20)"
Numerical Value	I lost my husband - this month will be a year on the 14 - so I'm a single parent. Ich habe meinen Mann verloren - diesen Monat wird es ein Jahr sein - also bin ich alleinerziehend.

Table 19: **Detector Output examples** for the WMT21 Winning System using the 1M WMT20 Monolingual-Evaluation set

K	Cases in K
100	0
1000	0
10000	0
100000	3

Table 20: **Number of flagged Error cases present in the K worst scoring translations, as scored by COMET-QE (Rei et al., 2020)**. The erroneous sentences are present very sparsely even in the lowest 100K scoring translations.

Threshold	Sentences	Cases
0.1	251,518	26
0.2	346,083	36
0.3	522,128	70
0.4	649,828	79
0.5	742,194	88

Table 21: **Number of flagged Error cases present in the translations with score less than the threshold, as scored by COMET-QE (Rei et al., 2020)**. The erroneous translations are present across a range of scores.

G Related Work

We situate SALTED among previous works in Behavioral testing for NMT along five dimensions in Table 22. The five dimensions reflect the operational properties of behavioral testing methods:

1. **Instance-Level**: A method operating at an instance-level requires only the source-translation instance for an error to be adjudged. Typically, Behavioral testing methods rely on input modifications to test the model for errors, thereby requiring the generation of new translations. A method which doesn't work at the instance-level is better suited for exploratory uses than for obtaining targeted error measurements over a given corpus.

2. **Specification-Based**: A specification-based method explicitly consumes specifications of correct model behavior. For example, Behavioral testing methods which rely only on consistency measures over translations generated on an input set do not consume an explicit output behavior specification and thereby are hard to translate into actionable measurements.

3. **Modularized**: A modular method allows for fine-grained measurements of specific error categories using the same method by separating the concerns of the error detection algorithm and the error type. For example, a method which is not modularized is hard to adapt to a new error type.

4. **High-Precision**: A high-precision method produces very few false-positives, ensuring that the generated measurements are trustworthy.

5. **Generative**: A generative method allows for the generation of new samples either for metamorphic testing or for data augmentation or error correction.

Table 22 shows that compared to existing behavioral testing methods, SALTED is more comprehensive, thereby allowing for variety of use cases.

Method	Instance-Level	Modularized	Specification-Based	High-Precision	Generative
SIT	x	x	x	x	✓
PatInv	x	x	x	x	✓
TransRepair	x	x	x	x	✓
RTI	x	x	x	x	x
SALTED	✓	✓	✓	✓	✓

Table 22: A comparison of existing Behavioral Testing Methods for NMT along five dimensions. The compared methods are: SIT (He et al., 2020), PatInv (Gupta et al., 2020), TransRepair (Sun et al., 2020) and RTI (He et al., 2021).