Carnegie Mellon University

Text Normalization for Speech Systems for All Languages

Athiya Deviyani, Alan W. Black
Language Technologies Institute

Text normalization

also known as standardization

Mapping non-standard words (symbols, numbers, abbreviations) to standard words (strings of characters that have a standard pronunciation).

Most text-to-speech systems are limited to standard words.

Original: It will cost $5 to buy 2lbs of apples

Normalized: It will cost five dollars to buy two pounds of apples

Text normalization of currency and measure. The colors in the original text correspond to the standard form in the normalized text.

Motivation

- Major degradations of perceived quality in Text To Speech systems can be traced to problems involving text normalization
- Previous work attempted the automatic mapping of non-standard words to standard words using statistical, neural, and rule-driven approaches, which work but **have poor generalizability to new languages**

Related work: Google Kestrel's TTS

- Classify non-standard tokens into their respective semiotic classes (**classification grammars**) and taking their context to appropriately verbalize the token (**verbalization grammars**)
- Classification and verbalization grammars are **compiled into weighted finite-state transducers** (WFSTs) which will be used to pass non-standard text into

Problem: the method requires complex hand-written grammars, which means that it is difficult to dynamically scale to multiple languages

Carnegie Mellon University

P. Ebden and R. Sproat, "The kestrel tts text normalization system," Natural Language Engineering, vol. 21, no. 3, pp. 333–353, 2015

Related work: FST-based verbalizers

- Built through the **efficient sourcing of language-specific data collected through questionnaires** given to native speakers of a language
- The **collected data contains all the necessary information to bootstrap the number grammar induction system** to parameterize a verbalizer template, which will then be used by formal language experts to develop FST-based text normalization systems

Our method takes in the idea of text normalization as a **data collection problem** and improves it further by providing a **user-friendly interface** and a feature which allows users (both the native speaker source and the formal language expert) to **easily amend the provided information** and the text normalization system for any obvious mistakes.

Carnegie Mellon University

S. Ritchie, R. Sproat, K. Gorman, D. van Esch, C. Schallhart, N. Bampounis, B. Brard, J. F. Mortensen, M. Holt, and E. Mahon, “Unified verbalization for speech recognition & synthesis across languages.” in INTERSPEECH, 2019, pp. 3530–3534.

Related work: NN-based methods

- Treat text normalization problem as a **machine translation task**, where the source language is raw text and the target language is normalized text (in the same language as the source language)
    - Use architectures such as the **recurrent neural network** (RNN), or the more recent **transformer**

Problem: deep learning-based methods are notoriously known to be “data hungry”, thus a large amount of non-standard and standard text pairs are needed. Therefore, any supervised learning-based text normalization method will not be transferable in the context of low-resource languages.

Carnegie Mellon University

H.Zhang,R.Sproat,A.H.Ng,F.Stahlberg,X.Peng,K.Gorman, and B. Roark, “Neural models of text normalization for speech applications,” Computational Linguistics, vol. 45, no. 2, pp. 293– 337, 2019.

Given access to a native speaker,

Given access to a native speaker, what are the most important questions we should ask them

Given access to a native speaker, what are the most important questions we should ask them in order to perform *basic counting* in their language,

Given access to a native speaker, what are the most important questions we should ask them in order to perform *basic counting* in their language, without having to write additional code?

Semiotic classes

Groups of non-standard words that occur in text. Semiotic classes are **distinguishable by the patterns** within the non-standard words.

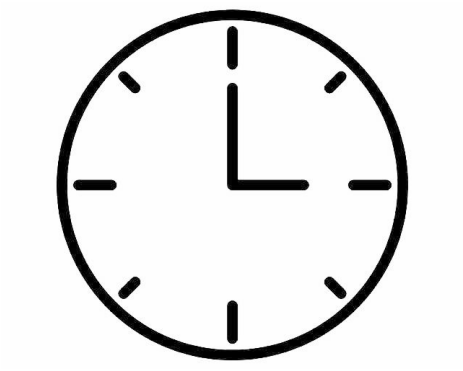
Semiotic class	Examples
Cardinal	1235523, 1,000,000
Time	19:10, 1:00PM
Decimal	21.25, 0.034
Digit	15213, 007
Measure	90%, 5lbs
Money	$23, 50.00USD

Carnegie Mellon University

Taylor, Paul. Text-to-speech synthesis. Cambridge university press, 2009.

Challenges

Standardization **within a semiotic class.**



"The time is now, twelve fifteen **AM**."

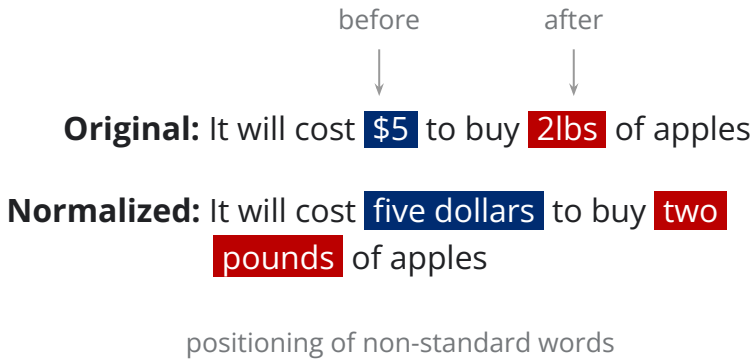
"It is now a **quarter** past midnight."

"Currently, it is fifteen **minutes** past twelve."

Carnegie Mellon University

Challenges

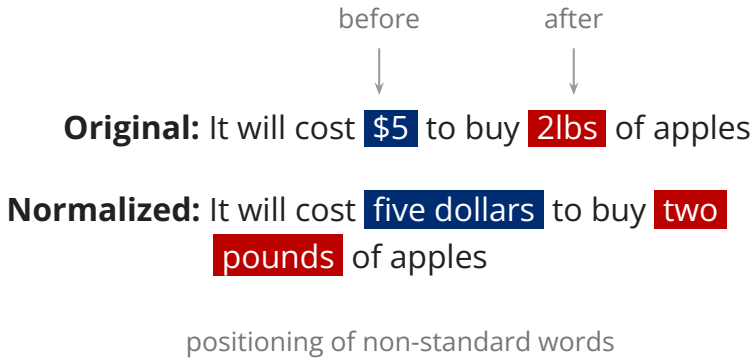
Standardization across **various semiotic classes.**

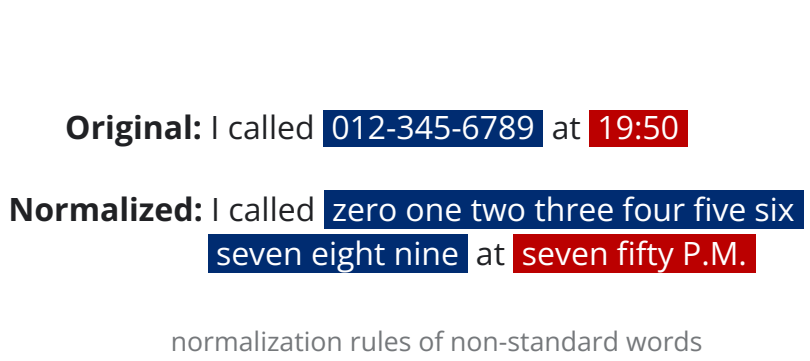


Carnegie Mellon University

Challenges

Standardization across **various semiotic classes.**





Carnegie Mellon University

Challenges

Standardization across **various languages** within the **same semiotic class**.

Carnegie Mellon University

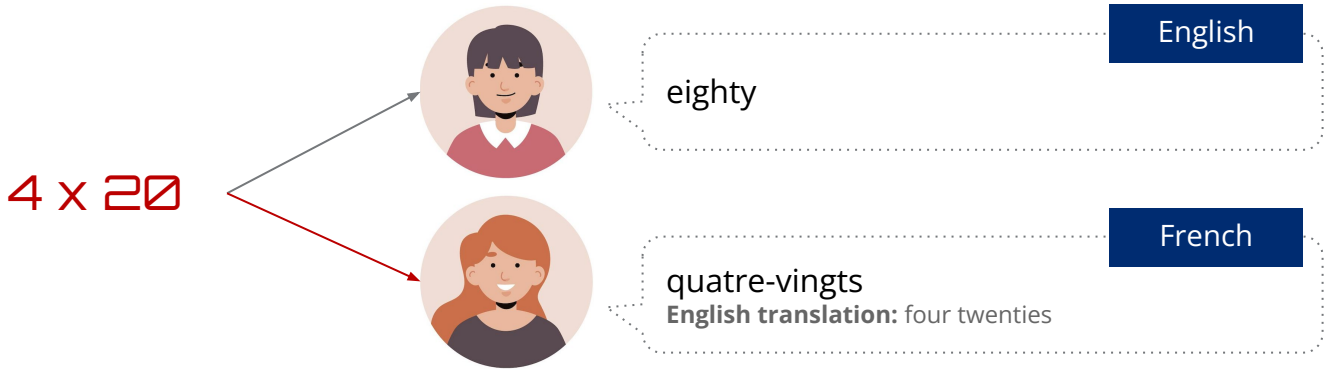
Challenges

Standardization across **various languages** within the **same semiotic class**.

Carnegie Mellon University

Challenges

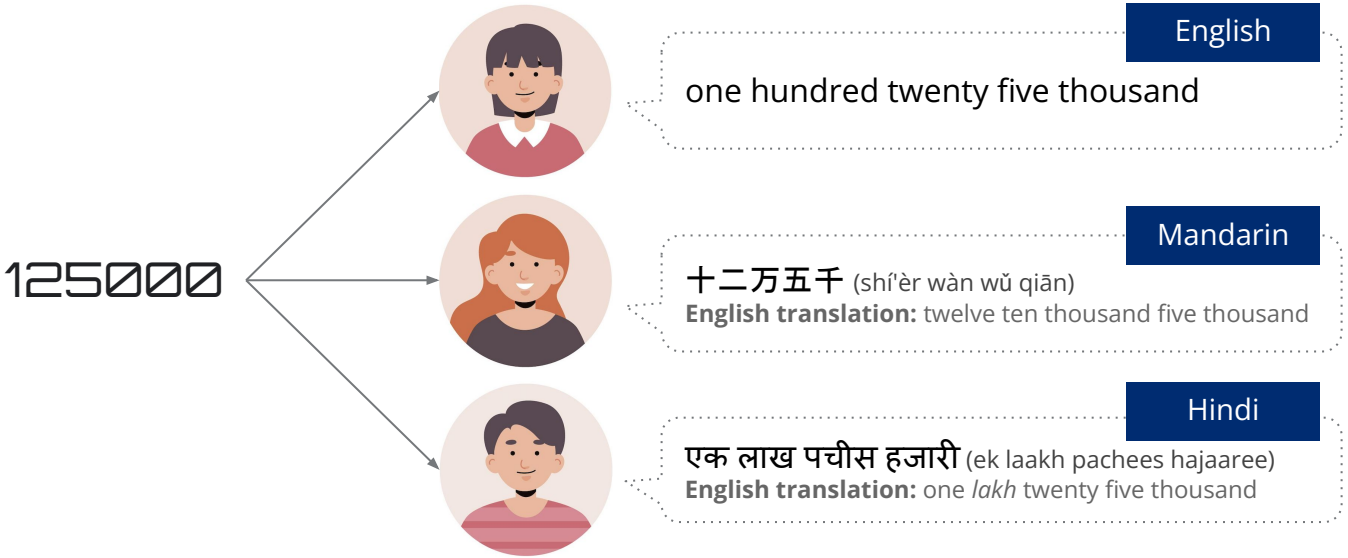
Standardization across **various languages** within the **same semiotic class**.



Carnegie Mellon University

Challenges

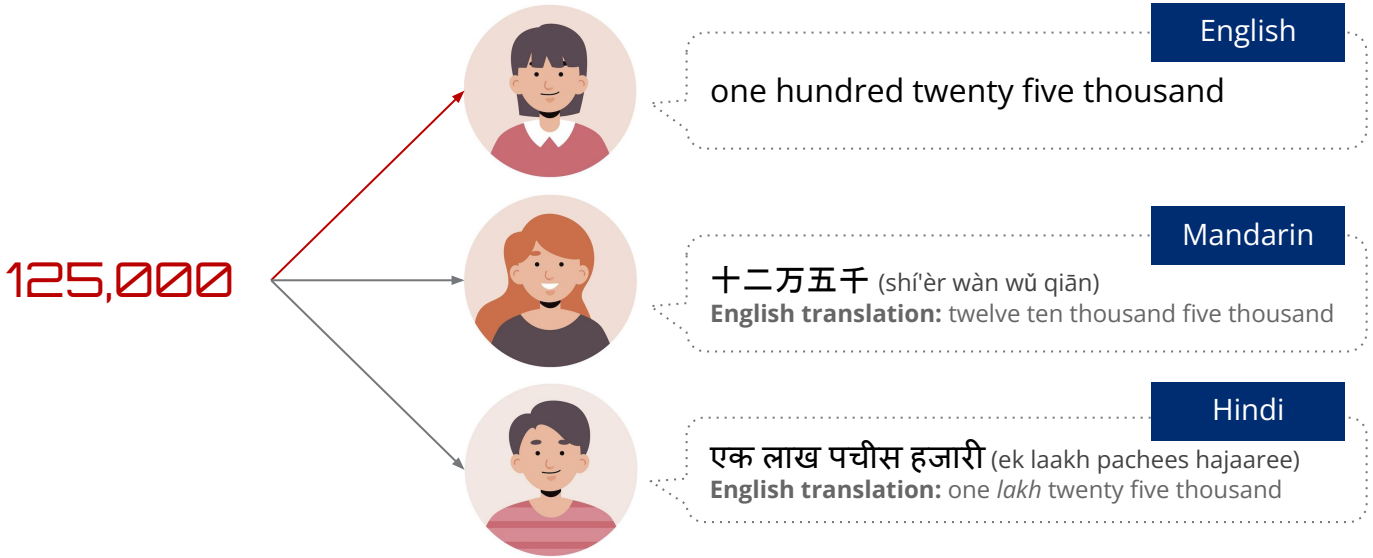
Standardization across **various languages** within the **same semiotic class**.



Carnegie Mellon University

Challenges

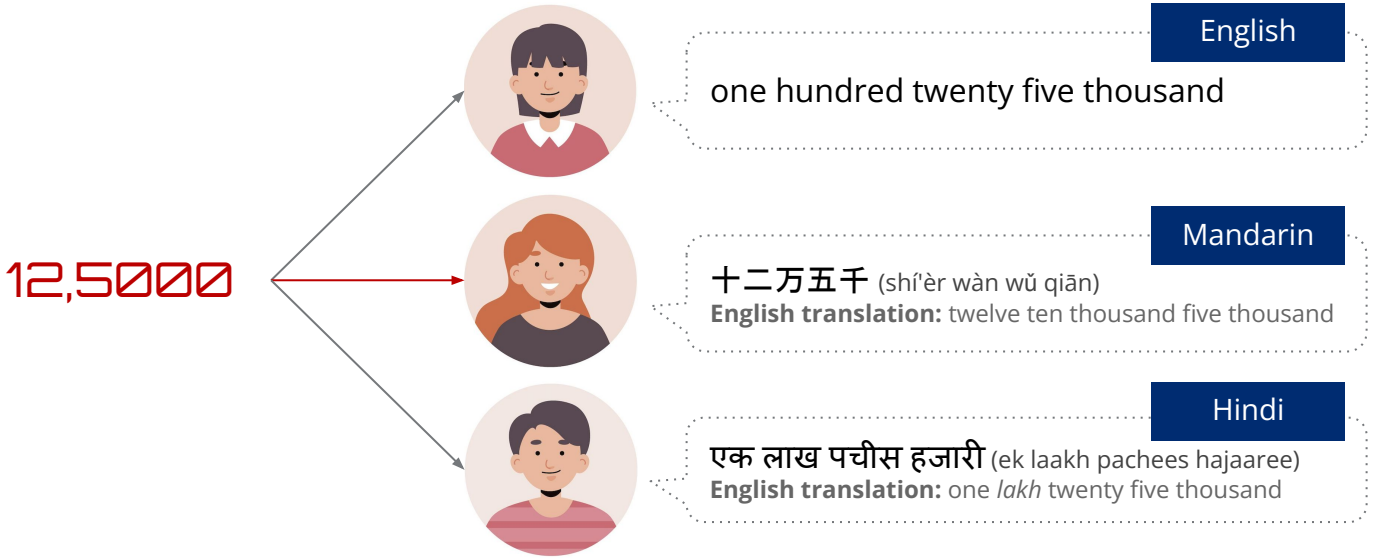
Standardization across **various languages** within the **same semiotic class**.



Carnegie Mellon University

Challenges

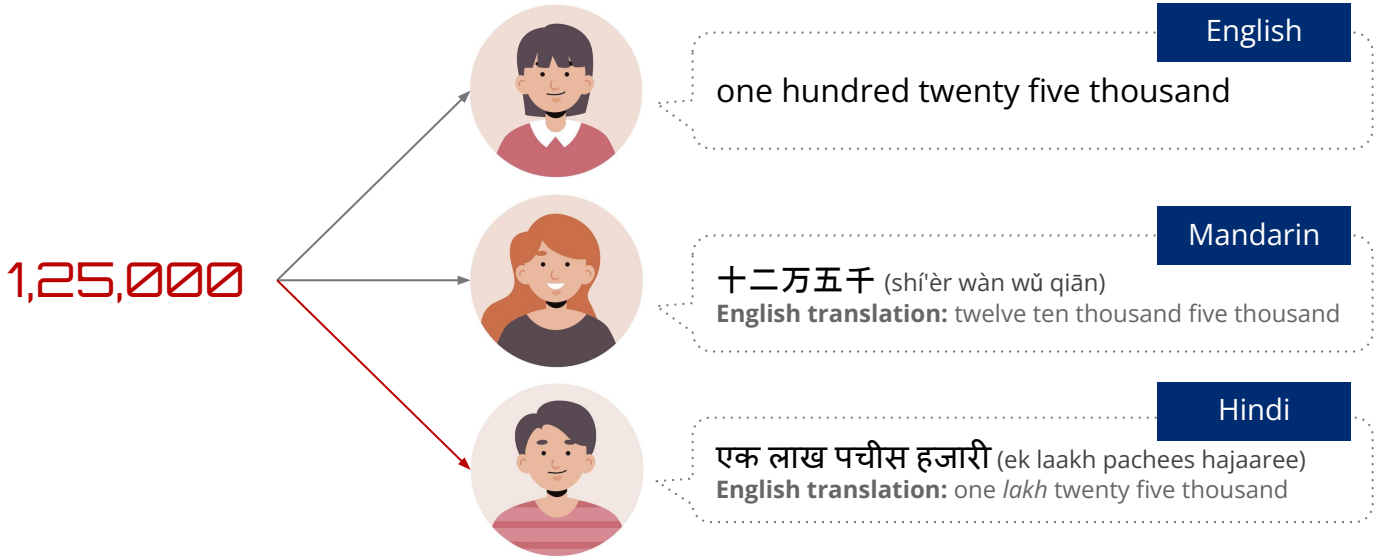
Standardization across **various languages** within the **same semiotic class**.



Carnegie Mellon University

Challenges

Standardization across **various languages** within the **same semiotic class**.



Carnegie Mellon University

Proposal

- A method to **collect data** regarding the counting system, digit mapping, and relevant information with respect to the different semiotic classes from **native speakers of a language (without any coding experience)** to allow for the development of a **generalizable** text normalization system

Proposal

- A method to **collect data** regarding the counting system, digit mapping, and relevant information with respect to the different semiotic classes from **native speakers of a language (without any coding experience)** to allow for the development of a **generalizable** text normalization system

Research questions

- **RQ1:** Which non-standard words appear in text?
- **RQ2:** How do we ask questions to get our answers?
- **RQ3:** How do we make it easy to answer such questions?
- **RQ4:** How do we address variability?
- **RQ5:** How do we extend this to a full front end?

Carnegie Mellon University

Methodology: Data Collection

- **Main goal:** obtain sufficient information from a native speaker in order to map non-standard numerical text to its standardized form, ideally by asking the minimum number of questions
- After a careful evaluation of the different counting systems adopted by several (primarily) high resource languages, we observe that the **nuances within/across cardinal numbers diminish as the number grows**
- Non-standard numerical texts that are commonly found in most corpora are usually **smaller numbers** (e.g. 1,000,000 in text would normally be printed as 1M or 1 million)

Methodology: Data Collection

- We decided that it is sufficient to collect information on the **digits 1-100**, as well as the **numbers denoting the tens position**, such as hundred, thousand, up to a billion
- We have requested some information from the user regarding the **additional symbols** that might exist in numerical text which are **telling of the different semiotic classes**, such as the decimal point (.), currency symbol and its normalized form ($ -> dollar), percentage sign (%), and more

Methodology: Data Collection

Carnegie Mellon University

Methodology: Text Normalization Algorithm

- When the user uploads the CSV file containing information on the non-standard to standard text mapping for a particular language, the back-end will **parse the information into a hashmap to be normalized using a Python script**
  - At the moment, the algorithm is able to normalize text in the form of whole numbers, phone numbers, currency, floating-point numbers, time, and percentages
- The algorithm takes into account how the numbers are **chunked** and **processes each chunk accordingly**
  - e.g. `process("123456") -> process("123") + process("456")`
- The algorithm will also **take into account the surrounding symbols** to identify the semiotic class by querying for specific symbols such as decimal points, colons, and more
  - e.g. `if currency_symbol in text: process_currency(text)`

Carnegie Mellon University

Methodology: Text Normalization Algorithm

- For **currencies**, **floating-point numbers** and **percentages**, the non-standard text will be **split depending on their position with respect to the decimal point**
  - The numbers before the decimal point will be read as a whole number, while the numbers after the decimal point will be read out individually.

    e.g. `process_currency($2.52) -> process(2.52) + process($) -> process(2) + process(52) + process($) -> two point five two dollars`

  - **Future work:** take into account stylistic preferences, e.g. two point five two dollars vs. two dollars and fifty two cents

Carnegie Mellon University

Methodology: Text Normalization Algorithm

- **Phone numbers** will be normalized into **individual numbers**, similar to the numbers behind a decimal point
  - e.g. `process(4122682000) -> process(4) + process(1) + … + process(0)`
  - **Problems:** stylistic preference varies across countries, e.g. placement of parentheses in US phone numbers to denote area code ((412) 268-2000), length of hotlines are usually short (911), so would need context clues
- Texts belonging to the **time** semiotic class are usually indicated by a colon in the middle of a string of digits (20:53), reading out the numbers before the colon, followed by a space, and then the numbers after the colon
  - e.g. `process(20:53) -> process(20) + pause + process(53)`
  - **Problems:** stylistic choices of using AM/PM if 12-hour time is preferred

Carnegie Mellon University

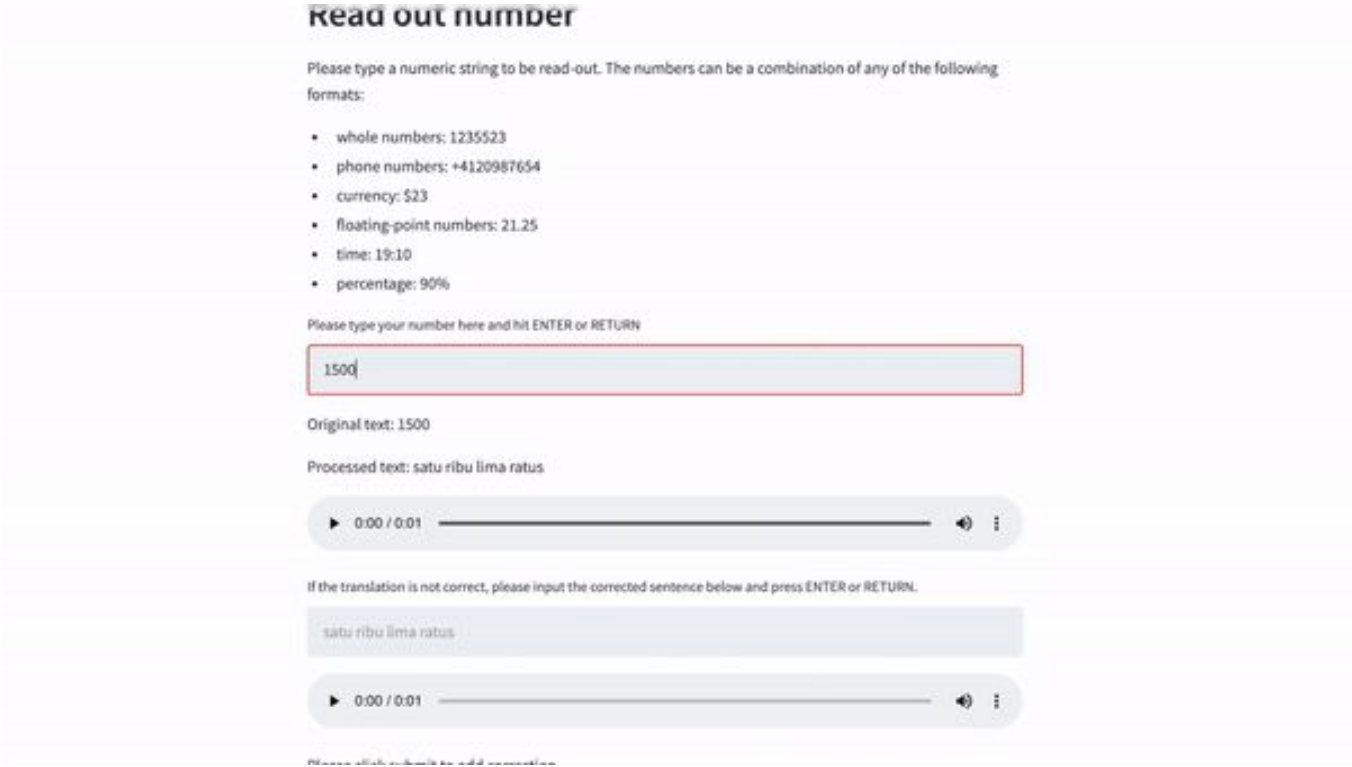
Methodology: Evaluation and Feedback

- As our text normalization system mainly adopts a rule-based methodology which is quite stringent and **might not generalize well** to languages with unique counting systems
- To aid this, we **provide a feedback collection system** where a user will be able to **correct an erroneous normalization** after the .wav file is generated and played on the website
- After the user has input the correct normalization, they will **be able to listen to the amended normalization** (generated by a Flite synthesizer) and they will be able to submit it once they are satisfied
- Given enough users and feedback from erroneous normalizations, this data will allow us to **identify common mistakes** that arise from the text normalization of specific languages and **make the appropriate amendments to the algorithm** or **add new fields to collect more information on the CSV file**

Carnegie Mellon University

A.W.Black and K.A.Lenzo, “Flite: a small fast run-time synthesis engine,” in 4th ISCA Tutorial and Research Workshop (ITRW) on Speech Synthesis, 2001.

Methodology: Evaluation and Feedback

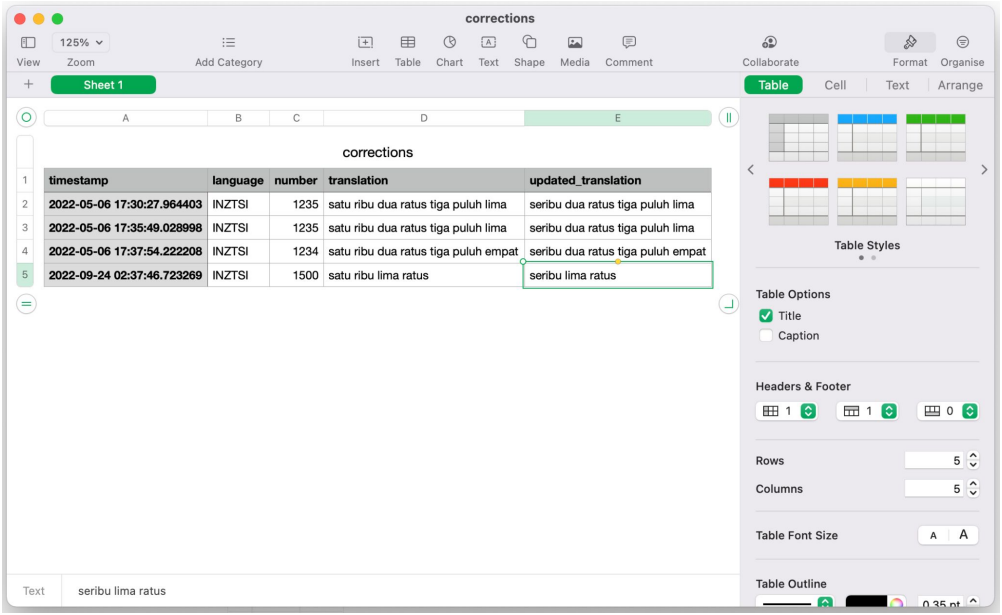


Carnegie Mellon University

Methodology: Qualitative Evaluation

Observation

For the Indonesian language, we can see that the errors have one thing in common: even though “1” is “satu” and thousand is “ribu”, the verbalization of 1000 is not “satu ribu” but “seribu”.



Carnegie Mellon University

RQ1: Which non-standard words appear in text?

- We have found that the type of **non-standard words** which involve numerical characters **can be grouped into different semiotic classes**
- The presence of a semiotic class highly **depends on the corpus** the text originates from
    - e.g. a science textbook will have multiple instances of measure words such as “180kg” or “2000m”, while a news article might contain date and time instances such as “20:30”
- There exists **ambiguities** between the existing semiotic classes, such as how “20:30” can represent an instance of time, but it can also indicate a verse from the Bible
- Without taking into account the context of the text, **the system will normalize the non-standard words into its most generic form** for generalizability

Carnegie Mellon University

RQ2: How do we ask questions to get our answers?

- Our proposed method **relies heavily on user input and feedback**, where a user will fill the required information in a table for the initial non-standard to standard text mapping, and then make amendments to erroneous normalizations
    - This will be used to **improve the algorithm** as well as to **evaluate whether we would need to gather more information** by adding fields to the table
- The **table will contain information such as the digit normalization** from 1-100, common symbols such as currency and percentage, decimal point, number chunking and more
- Given access to a native speaker of a low-resource language, this process is **relatively easy and efficient**
- The evaluation process will involve primarily generating non-standard text from the various semiotic classes and calculating the number of correct and incorrect normalizations to obtain an accuracy score

Carnegie Mellon University

RQ3: How do we make it easy to answer such questions?

- We present an accompanying **web-app** to our tool which was designed with intuitiveness in mind
- Through the website, a native speaker with no prior programming skills will be able to **input the non-standard to standard mapping of various words** such as digits, currency, tens, mathematical symbols and more, by filling in a spreadsheet in the form of a comma-separated value (CSV) file
- The back-end of the website will then **automatically parse the spreadshee**t, allowing the user to read and listen to the normalization of the non-standard text and **provide any feedback or corrections**

Carnegie Mellon University

RQ4: How do we address variability?

- Given that our system hosts over 600 unique languages on a public server for public use, the data we are collecting is **vulnerable to malicious or erroneous input**
    - **Prevention:** store multiple tables from various languages and we will take the most common mapping for each item. This will also solve the issue of dialectal variation that may arise from the variety of input collected from the native speakers.
- **Disambiguation:** taking the most common mapping for each item within a language

RQ5: How do we extend this to a full front-end?

- There are still **plenty of non-standard text that may appear** in a corpus which **do not include numerical values**
  - e.g. alphabetical sequences which are unseen but pronounceable, letter sequences such as acronyms and organization names (e.g. CIA, WHO) whose letters are to be read individually, or shortened form/abbreviations which are commonly used (e.g. dept → department)
- **Extension:** capture these nuances from multiple languages using the same data collection process

Carnegie Mellon University