Data processing for Japanese text-to-pronunciation models

Gleb Mazovetskiy
Google
glebm@google.com

Taku Kudo
Google
taku@google.com

Presentation by Gleb Mazovetskiy at JLR2024

Text-to-pronunciation

1. Take any text as input.

2. Produce the pronunciation of this text, e.g. as hiragana.

For Japanese, this is a **very hard** problem to solve!

Gleb's 1-10 scale for text-to-pronunciation task difficulty:

Spanish: **1/10**. Easy. Mostly orthographically transparent.
English: **3/10.** Medium. Some ambiguity, mostly homographs and G2P.
Hebrew: **5/10.** Hard. Vowels not indicated.
Japanese: **10/10**.

Many readings

Kanji have 2-6 common pronunciations that depend on meaning / context.

3月1日は日曜日で祝日　晴れの日でした

sangatsutsuitachiwanichiyobideshukujitsu harenohideshita

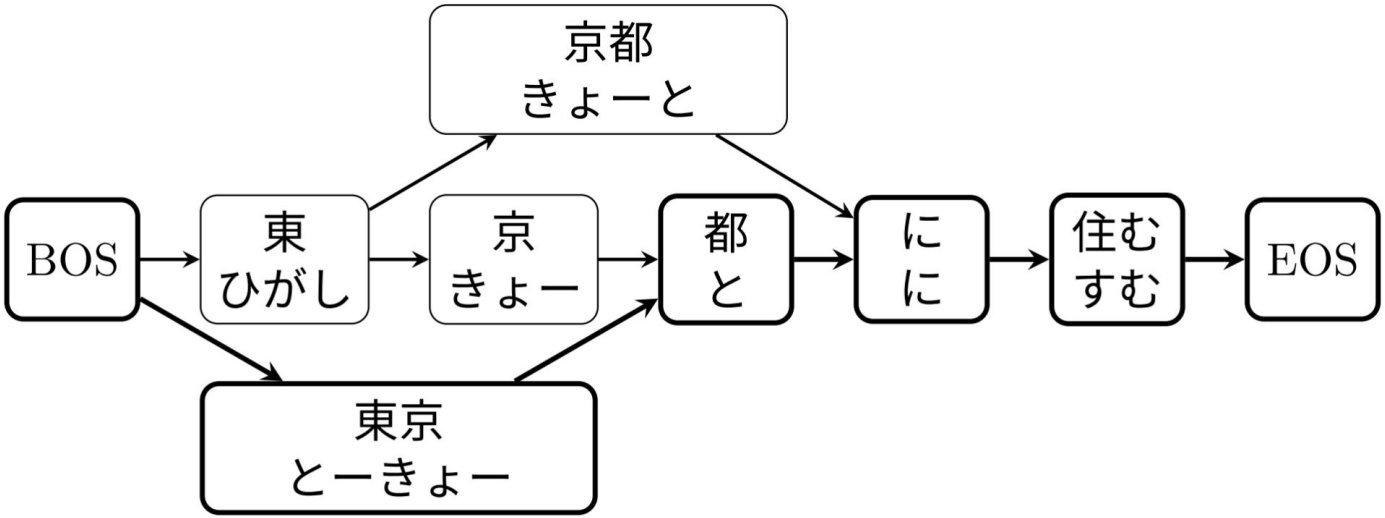
March 1st was Sunday, it was a national holiday and a sunny day

Words are not separated by spaces.

No single "best" or "golden" standard for segmentation.

Segmentation & pronunciation

Possible segmentations and pronunciations are interdependent.



Segmentation & pronunciation lattice for "東京都に住む" (abridged)

Segmentation & pronunciation (cont.)

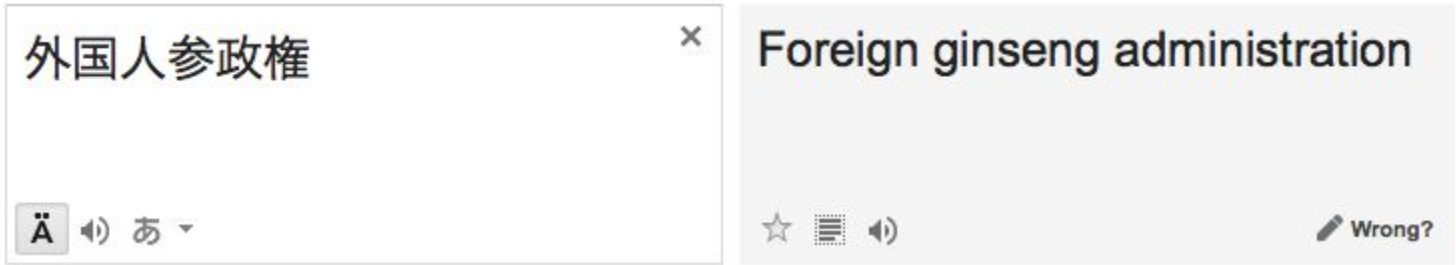
Mistakes often change the meaning completely.

外国人｜参政｜権 → **OK** *(right of foreigners to vote)*

gaikokujin *(foreigners)* | sansei *(vote)* | ken *(right)*

外国｜人参｜政権 → **NG** *(foreign carrot administration)*

gaikoku (*foreign*) | ninjin *(carrot)* | seiken *(administration)*



Named entities

Named entity pronunciations are often arbitrary and highly irregular.

- Person names
  - 羽生 結弦 Hanyu Yuzuru (figure skater)
  - 羽生 善治 Habu Yoshiharu (politician)
- Place names
  - 三田駅 Mita-eki in Tokyo but Sanda-eki in Hyogo
  - 渋谷 Shibuya or Shibutani, both are in Tokyo
- Highly irregular: kirakira names, brand names, etc.

Even native speakers don't always know how to read something.

Semiotic classes

Semiotic classes are numbers, dates, monetary amounts, etc.

Example: 110

ひゃくとー	hyaku-tō
いちいちぜろ	ichi-ichi-zero
いちいちれー	ichi-ichi-rei
わんわんぜろ	wan-wan-zero
ひゃくじゅー	hyaku-jū

Pitch accents

Pitch accents are low/high tones with stress that apply to subphrases rather than syllables.

Pitch accents carry meaning and have highly irregular sandhi.

端	^はし	edge
箸	^は!し	chopsticks
橋	^はし!	bridge

(^ is the accent phrase start, ! follows the down-pitch position)

More on this later.

How do we solve this?

With **lots** of data!
However:

1. Varied annotation standards.
2. Some data is only partially annotated:
   only some words, no pitch accents, etc.

This talk is about techniques for ingesting all kinds of data!

Techniques developed by us for the Japanese text-to-pronunciation model used by Google in production since 2020 for text-to-speech and other tasks.

Readings vs Pronunciations

とうきょう
東京

In Japanese dictionaries and corpora, the correct pronunciations are often annotated with "readings" (e.g. furigana).

However, readings are not pronunciations!

Consider:

Spelling	Reading	Pron
工藤	くどう ku-do-u	くどー
武井	たけい ta-ke-i	たけい (not たけー) ta-ke-i (not ta-kē)
女王	じょおう jo-o-u	じょおー (not じょーう) jo-ō (not jō-u)

Readings vs Pronunciations model

Let's build a model to map readings to pronunciations!

Observation: Vowel elongation cannot happen at the start of a kanji pronunciation but should happen everywhere else, with a few exceptions*.

Solution: Obtain per-Kanji character reading using unsupervised learning (EM)

* Exceptions are mainly Jukujikun, such as 今日 (きょー).

Readings vs Pronunciations model (cont.)

Training: For each dictionary word, build a lattice of all kanji-reading pairs.

Example: 武井

surface: 武 井

reading: た | け | い (mora segmentation)

Possible alignments under the reading constraint:

武 : た | たけ

井: けい | い

Best alignment: 武 井 → たけ | い (no vowel elongation in けい)

Find the best alignments with unsupervised EM algorithm
(IBM1 algorithm with linear-alignment constraint)

Fuzzy matching

We can now convert all readings to pronunciations! 🎉

However, the readings themselves are also not standardized.

For example, the word グアバ (guava) may be transcribed as:

ぐぁば (gua-ba, 2 moras)
ぐわば (gwa-ba, 2 moras)
ぐあば (gu-a-ba, **3** moras)

These 3 readings are transcribed differently but they describe the same pronunciation (or nearly so) of the word グアバ in practice.

A system that can ingest all kinds of data should be able to handle all of these!

Fuzzy matching: Sources of fuzziness

1. Vowel elongation: ああ ≈ あぁ ≈ あー
2. Diphthongs (aka yōon): りょ ≈ りよ
3. Historical kana orthography: え ≈ ゑ、が ≈ ぐわ (still in use in named entities)
4. Yotsugana: じ ≈ ぢ、ず ≈ づ (for dialects with no distinction)
5. Some other cases: ヴ ≈ ぶ、いう ≈ ゆー、....

Combination of the above: ばあ ≈ ばぁ ≈ばー ≈ ゔぁあ ≈ ゔぁぁ ≈ ゔぁー

Some of these may sometimes be used represent slightly different sounds, while others are always identical.
Equivalence definition depends on the downstream task.

Fuzzy matching model

1. Over 100 sets of fuzzy-equivalent prons, hand-built by Taku.

2. Fuzzy alignment algorithm. For models that explicitly constrain the output space, e.g. with a lattice, we want to align the corpus annotation to the output space while propagating pitch accents.

corpus annotation	lattice node	result
^けづぃ!ん	けびん	^けび!ん
^ぐぁ!ば	ぐあば	^ぐあ!ば - accent mora changed from 1 to 2
^ぐ!あば	ぐぁば	🚫 cannot align accent position: ぐぁ is a single mora.
^い ^ち	いち	🚫 cannot propagate accent start boundary

Segmentation differences

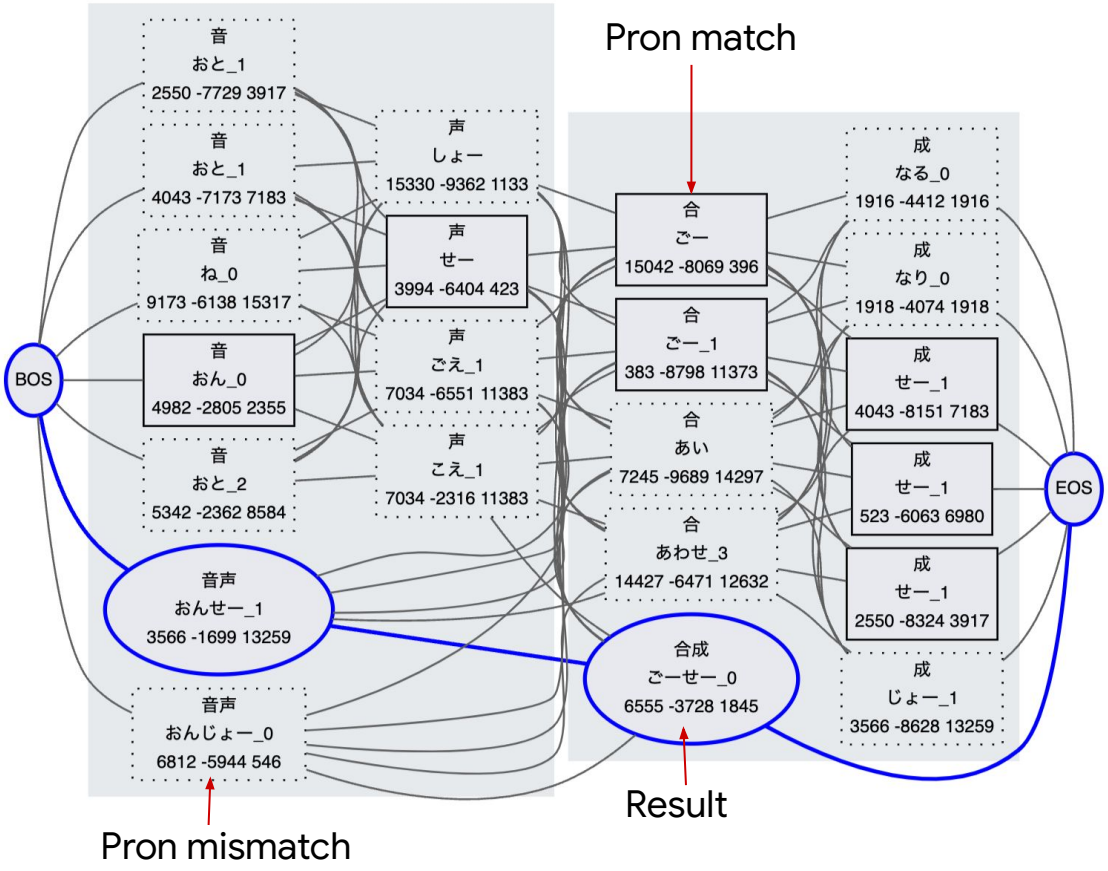
How do we train from corpora that are segmented into words using different annotation standards, or not segmented at all?

Observation 1: Segmentation granularity often does not matters for downstream tasks.

Observation 2: Segmentation in the training data should be consistent for models that produce some segmentation.

Solution: Resegment everything with another model **under the annotated pronunciation constraint**, i.e. the resulting segmentation must not prevent the annotated pronunciation.

Resegmentation with MECAB under pronunciation constraint



Annotated example:

音声合成 (voice synthesis) →
おんせーごーせい (onsē gosei)

Align example pronunciation to the mecab lattice using the fuzzy matching algorithm.

MECAB chooses the best path among the allowed nodes.

What model to use?

Now that we have the data, why not just use a sequence-to-sequence model?

You probably can, especially if starting of with a large multi-modal model!

However:

1. Training seq2seq from partially annotated data is tricky (lots of our data).
2. Unconstrained seq2seq models are prone to "silly errors".
3. Model size, training time, and maintenance cost (ease of bugfixing) can be a concern.

Mecari: Google's Japanese text-to-pronunciation model

Mecari, the model we use at Google avoids all of these issues:

1. Avoids many types of "silly errors" by design.
2. Fast to train (~30m).
3. Runs quickly even on a slow CPU (e.g. low-end phone).
4. Small enough to run on-device (e.g. your phone).

Mecari powers all the Japanese text-to-speech at Google since 2020 both on the server and (pruned and quantized) on-device.

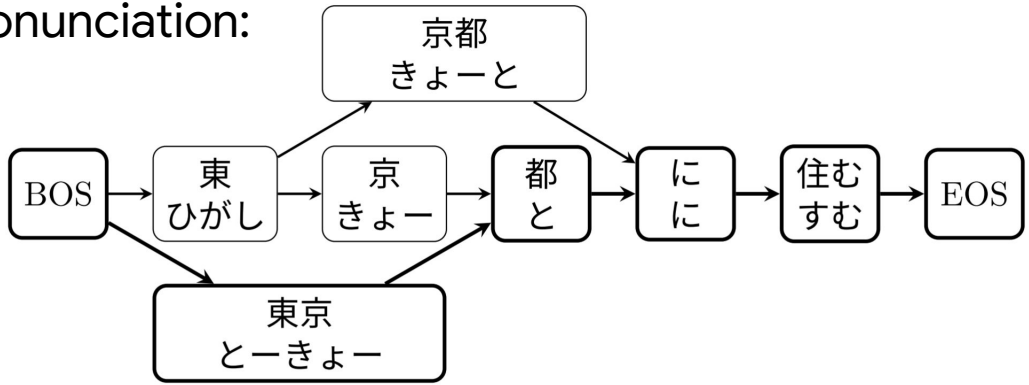
How does it work?

Mecari: Lattice construction

The traditional pipeline approach is to segment the input into words first, then resolve ambiguities.

This is suboptimal for Japanese because segmentation and pronunciation resolution is a **joint problem**.

In Mecari, we build a lattice with all the lexical and generated entries, such as semiotic classes, G2P for foreign scripts, etc. Every node contains a pronunciation:



Each path through the lattice represents a pronunciation of the entire input.

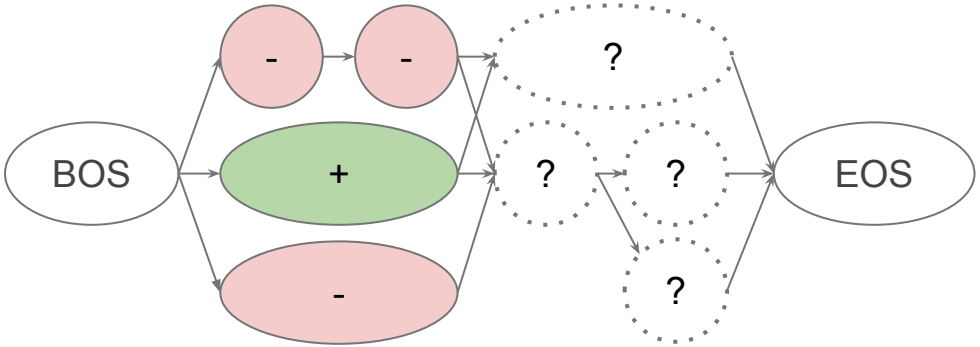
Mecari: Node scoring

Each node is assigned a score using an ML model.

Node are scored independently of each other.
The scoring model has access to the node, the input, and the lattice.

Scoring nodes independently allows training from **partially annotated data**.

Training example:

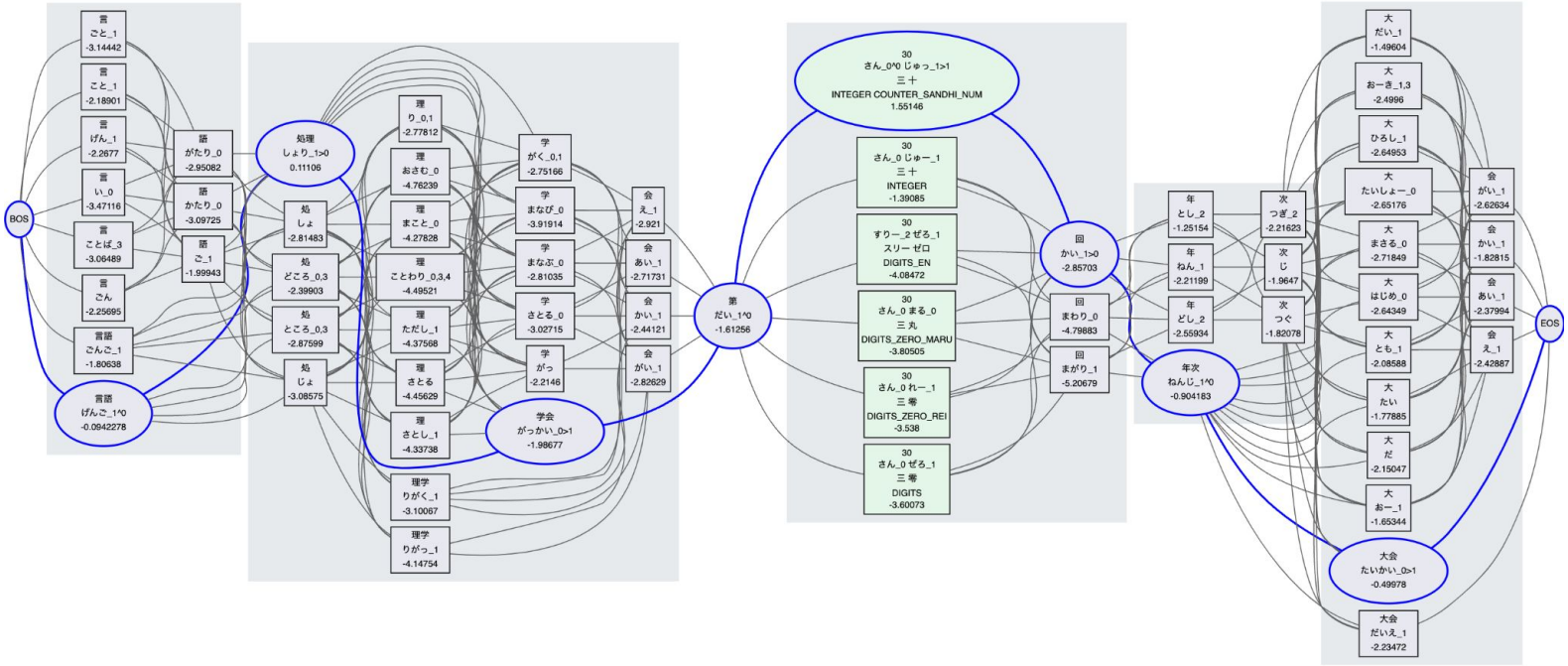


Here, the "+" node is the annotation match, "-" nodes are mismatches.
"?" nodes are unannotated, we do not propagate gradients to them when training.

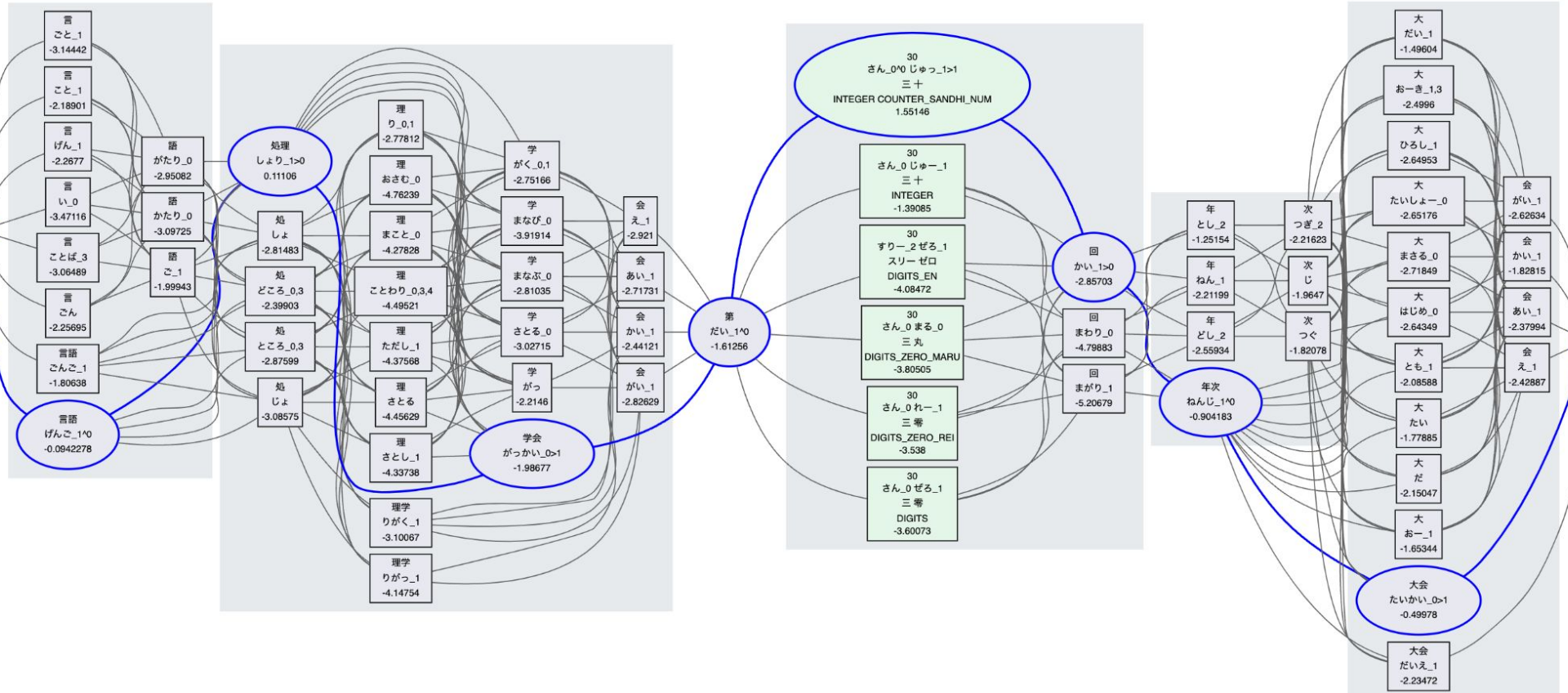
Mecari: Finding the lowest-cost path

Once the nodes have been scored, Mecari finds the best path using Viterbi search.

Example lattice for "言語処理学会第30回年次大会" after Viterbi search.



Viterbi search is $O(n^2)$ at worst. It is fast in practice due to the large number of "cutpoints".



Mecari: Pitch accent assignment

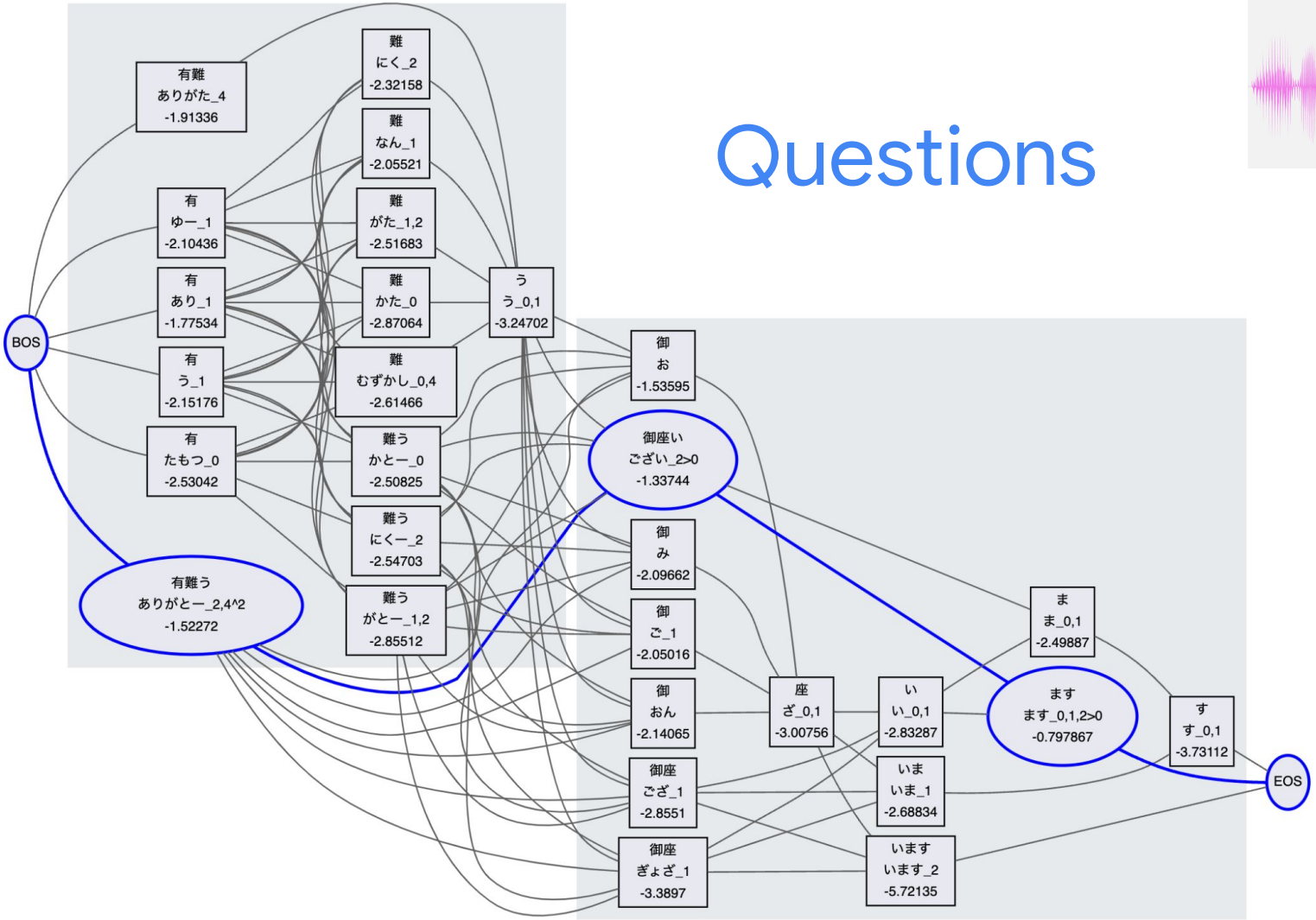
Once the best path has been found, we assign pitch accents using a separate model. The pitch accent assignment model also has access to the lattice structure.

We plan to publish more details about Mecari later this year.

The particular model that we use is an implementation detail.
Any scoring model can be used as long as it scores the nodes **independently** from each other. This is what allows training from partially annotated data.

Currently, the scoring model that we use is a large linear model with hand-designed features, trained using a structured perceptron.

Questions


BOS
有難
ありがた_4
-1.91336
有
ゆー_1
-2.10436
有
あり_1
-1.77534
有
う_1
-2.15176
有
たもつ_0
-2.53042
有難う
ありがとー_2,4^2
-1.52272
難
にく_2
-2.32158
難
なん_1
-2.05521
難
がた_1,2
-2.51683
難
かた_0
-2.87064
難
むずかし_0,4
-2.61466
難う
かとー_0
-2.50825
難う
にくー_2
-2.54703
難う
がとー_1,2
-2.85512
う
う_0,1
-3.24702
御
お
-1.53595
御座い
ござい_2>0
-1.33744
御
み
-2.09662
御
ご_1
-2.05016
御
おん
-2.14065
御座
ござ_1
-2.8551
御座
ぎょざ_1
-3.3897
座
ざ_0,1
-3.00756
い
い_0,1
-2.83287
いま
いま_1
-2.68834
います
います_2
-5.72135
ま
ま_0,1
-2.49887
ます
ます_0,1,2>0
-0.797867
す
す_0,1
-3.73112
EOS