Group 12

Hallucination Span detection

Problem Overview

Find out what part of this LLM response is a hallucination.

When was "Animals" released?	"Animals" was released by Prince in 1977.

This span is hallucinated (false)

Models in use

- Fact-checking models
  - Retrieval-based methods to verify against external data
  - FEVER dataset, Google Fact Check
- Consistency Models
  - Looks for internal inconsistencies
  - T5 - Google News summary
- Pre-trained Neural Nets
  - WordTune Spice – fact providing writing assistant
  - Transformer-based LLM

Our architectures

- **Unsupervised GPT Prompting**
  Straight up asks GPT-4 for spans
- **SVMs**
- **Supervised mBERT Sequence Classifier**
  Trains NN on tokenized query-response pairs

Data at hand

- 499 rows
- 18 languages: Arabic, Finnish, French, German, Hindi, Italian, Swedish, and Chinese

query	response	label
When was "Animals" released?	"Animals" was released by Prince in 1977.	[2,3]

Automatic Hallucination Detection in Model Outputs

Leveraging GPT-4

Implementation Overview

Automate hallucination detection for multilingual data.

- **Key Tools Used**:
  - GPT-4o Mini for detection.
  - JSON format for structured outputs.

Zero-Shot GPT-4o

Prompts:

- You are an assistant tasked with finding errors in responses.
- The question was "{input}", and the response was "{output}". Identify the parts of the answer that are incorrect or unsupported by the question. Answer with only the incorrect spans in the answer as a list of token ranges (start and end indices), like in this example: "[(0,3),(8,20)]".

- Very weak results
- Tokens often outside string range
- Spaces between words or punctuation returned

Summary and Future Work

Key Findings

- **Challenges with GPT-4o Mini**:
  - Results were **unsatisfactory** for hallucination detection.
  - High variability and inconsistent outputs across languages.
- **Outcome**: Current approach is not reliable for automatic annotation in multilingual data.

Input features for mBERT and SVM

- Full response sequence (as opposed to single tokens)
- Query for context (is this a must?)
- Part-of-Speech tag
  - Used by us:
    - UPOS – Universal POS categories
    - XPOS – Language-specific details

More ideas:

- Named entity recognition
  - Very similar task – finding factual data that's potentially wrong
- Contextual features – abrupt topic changes

Why recall over precision?

Consider a lie detector …

Brushing off a lie is worse than being a little too suspicious of the truth.

Feature Learning

Support Vector Machine

Data preparation for SVM sequence classification

- tokenize & lemmatize

When was "Animals" released?	"Animals" was released by Prince in 1977.	[2,3]
where banana?	on table	[0, 0]

- Hallucination spans to binary token-labels

[when] [is] [animal] [release] [?]	[animal] [is] [release] [by] [Prince] [in] [1977] [.]	[0, 0, 1, 1, 0, 0, 0]
where banana?	table	[1]

- Concatenate input data (incl. Support tokens)
- Create "ignore labels"

  CrossEntropy ignores labels of -100

[CLS] when is animal release ? [SEP] animal is release by prince in 1977 .	[-100, ..., -100, 0, 0, 1, 1, 0, 0, 0]
[CLS] where banana ? [SEP] table	[-100, ... , -100, 1]

SVM Model training

Tokenization

- Used the same mBERT tokenizer as the main deep learning model for comparability
- -100 tokens for placeholders

Training

- Use sklearn SVC
- Mask –100 tokens
- Balanced class weights

Axes of experimentation in training

What we've tried

- In/exclude query
- In/exclude POS tokens
- Training params:
  - Kernel
  - C
  - class_weight

More ideas to try out

- Different tokenizers
- Data augmentation through translation
- Data balancing (?)
- More sources of knowledge

  e.g. taking votes of multiple classifiers

Evaluation of SVM model

Results

- Best parameters
  - Kernel: rbf
  - C=0.5
  - Balanced class weights
- Low recall (0.36) and precision (0.46)

Conclusions

- SVM may not be good at finding contextual relationships between tokens
- Maybe a different tokenizer would return better results
- Query and POS tokens should be included
- Low recall, meaning model is not good at finding hallucinations

Maybe a deep learning solution will generalize better

Deep Learning

Supervised mBERT Sequence Classifier

Data preparation for sequence classification

When was "Animals" released?	"Animals" was released by Prince in 1977.	[2,3]
where banana?	on table	[0, 0]

- tokenize & lemmatize
- Hallucination spans to binary token-labels

[when] [is] [animal] [release] [?]	[animal] [is] [release] [by] [Prince] [in] [1977] [.]	[0, 0, 1, 1, 0, 0, 0]
where banana?	table	[1]

- Concatenate input data (incl. Support tokens)
- Create "ignore labels"

  CrossEntropy ignores labels of -100

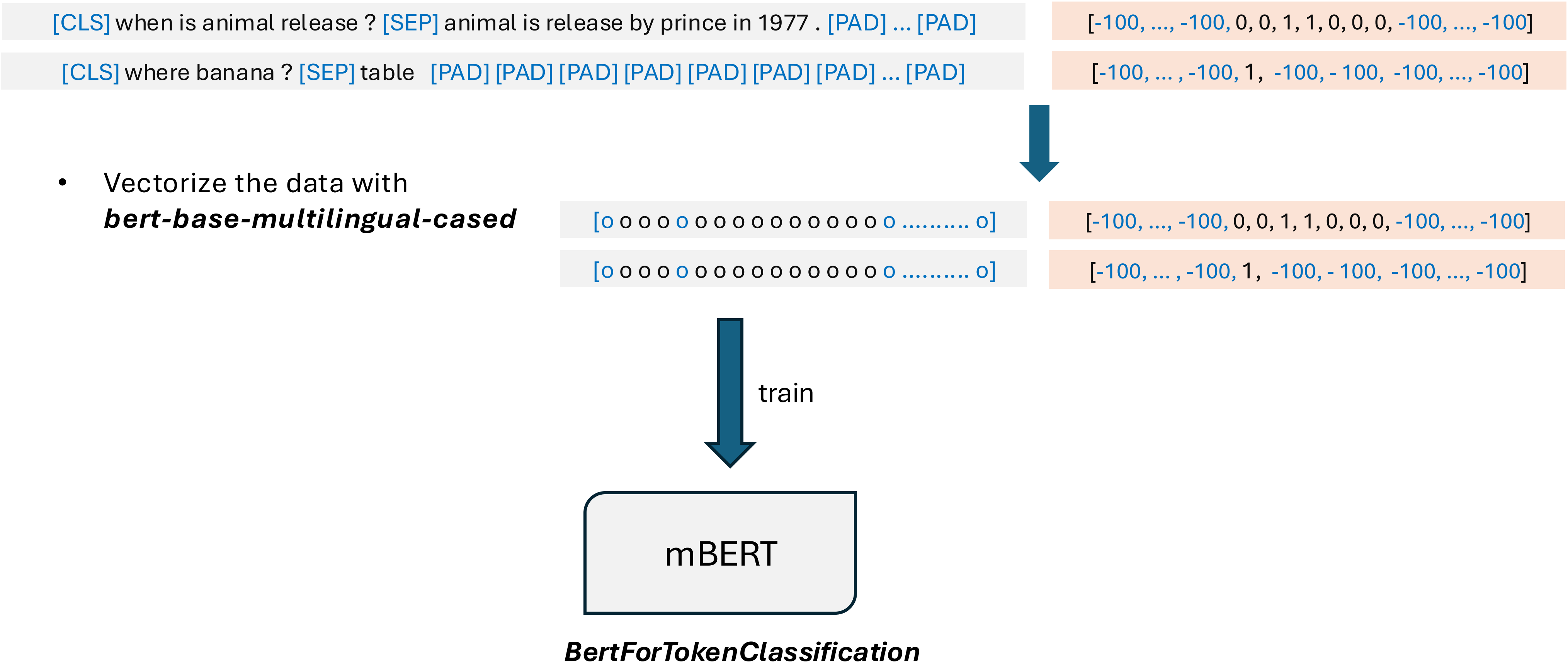
[CLS] when is animal release ? [SEP] animal is release by prince in 1977 .	[-100, ..., -100, 0, 0, 1, 1, 0, 0, 0]
[CLS] where banana ? [SEP] table	[-100, ... , -100, 1]

- Pad the data to fixed length

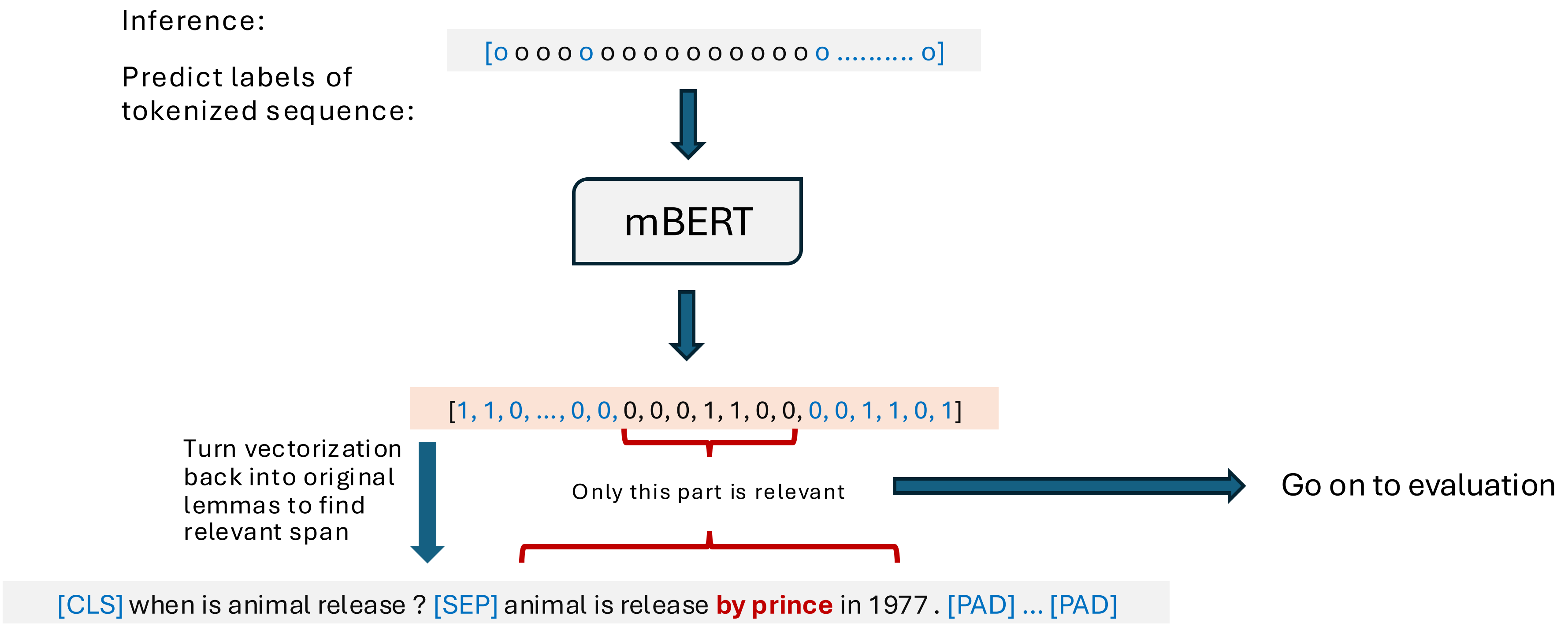
[CLS] when is animal release ? [SEP] animal is release by prince in 1977 . [PAD] ... [PAD]	[-100, ..., -100, 0, 0, 1, 1, 0, 0, 0, -100, ..., -100]
[CLS] where banana ? [SEP] table [PAD] [PAD] [PAD] [PAD] [PAD] [PAD] [PAD] ... [PAD]	[-100, ... , -100, 1, -100, - 100, -100, ..., -100]

Training a model with the prepared data

- Vectorize the data with ***bert-base-multilingual-cased***



Model application



Participating in the MUSHROOM Challenge?

Challenge

Original Text	Character labels	Tokens	Token labels
"Animals" was released by Prince in 1977.	[[22,31]]	", animal, ", ist, release, by, Prince, in, 1977, .	[[5, 7]]
Green Floyd consisted of 5 members.	[[0, 5], [25,26]]	green, floyd, consist, of, 5, member, .	[[0, 1], [4, 5]]

preprocessing

Number of characters changed.
Pattern matching with original strings?

No way back?!

Axes of experimentation

What we've tried

- In/exclude query
- In/exclude POS tokens
- Skip or truncate overflowing sequences
- Max length of encodings
- Training params:
  - Learning rate
  - Batch size
  - Patience
  - Max epochs
  - Different optimizer

More ideas to try out

- Different loss calculation (weighted?)
- Different tokenizers
- Data augmentation through translation
- Data balancing (?)
- More sources of knowledge

  e.g. taking votes of multiple classifiers
- Different pretrained models
  - Multilingual
  - Separate models per language

Excluding the query?

When was "Animals" released?	"Animals" was released by Prince in 1977.	[2,3]
where banana?	on table	[0, 0]

[animal] [is] [release] [by] [Prince] [in] [1977] [.]	[0, 0, 1, 1, 0, 0, 0]
table	[1]

[CLS] [SEP] animal is release by prince in 1977 .	[-100, -100, 0, 0, 1, 1, 0, 0, 0]
[CLS] [SEP] table	[-100, -100, 1]

[CLS] [SEP] animal is release by prince in 1977 . [PAD] ... [PAD]	[-100, -100, 0, 0, 1, 1, 0, 0, 0, -100, ..., -100]
[CLS] [SEP] table [PAD] [PAD] [PAD] [PAD] [PAD] [PAD] [PAD] ... [PAD]	[-100, -100, 1, -100, - 100, -100, ..., -100]

Including POS tokens

UPOS: universal (NOUN, VERB, ADJ, ADV, PRON, ADP, CONJ)

XPOS: language-specific (e.g. in english NN is a noun and NNs is a plural noun)

[CLS] where banana ? [SEP] on table	[-100, ..., -100, 1]
[CLS] where ADV WRB banana NOUN NN ? PUNCT . [SEP] on ADP IN table NOUN NN	[-100, ..., -100, 1, -100, -100, 1, -100, -100]

Optimizer ignores POS tokens

Supervised mBERT: what to do with overflows?

What if this is the max length??

[CLS] when is animal release ? [SEP] animal is release by prince in 1977 .

[-100, ..., -100, 0, 0, 1, 1, 0, 0, 0]

Options for handling overflow:

- Ignore and train anyway
  → Encoder will truncate the data!
- Skip observation entirely

Results

Average of all rows' sequences

All sequences concatenated

Number of observations after skipping

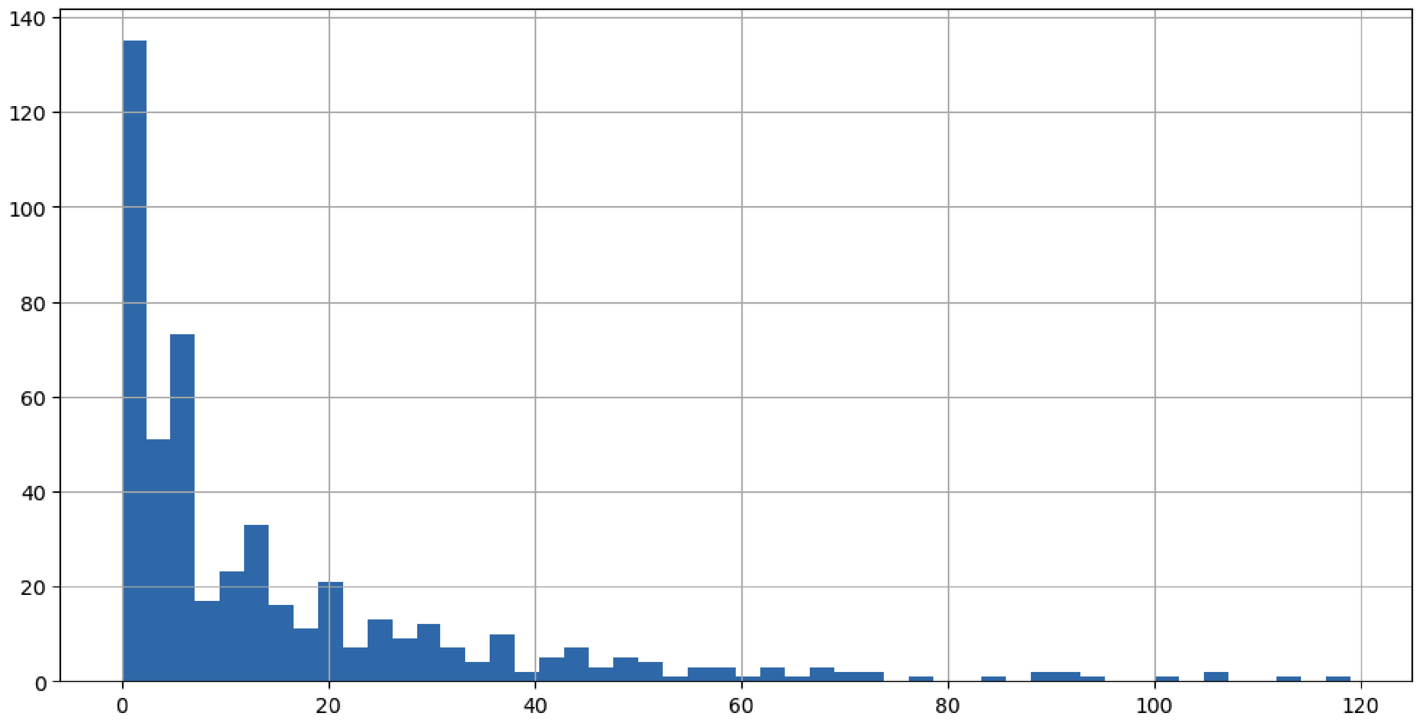
Number of tokens in sequence

Model	# Recall	# Recall flat	# Accuracy	# Accuracy flat	Overflow handling	N	Query included	POS tokens included	Max length	Patience
mBERT	**0,674**	**0,705**	0,777	0,740	Skip	479	X	X	512	3
mBERT	0,666	0,658	0,766	0,710	Truncate	499			512	3
mBERT	0,664	0,636	0,762	0,698	Truncate	499			512	10
mBERT	0,639	0,604	0,763	0,694	Skip	479	X		512	10
mBERT	0,637	0,660	0,778	0,742	Skip	479	X	X	512	10
mBERT	0,631	0,620	0,761	0,695	Skip	479			512	10
mBERT	0,614	0,630	0,772	0,735	Truncate	499	X	X	512	10
mBERT	0,601	0,647	0,779	0,759	Skip	385	X	X	256	10
mBERT	0,577	0,642	0,749	0,718	Skip	479		X	512	10
mBERT	0,548	0,576	0,777	0,754	Skip	479	X	X	256	3
mBERT	0,426	0,583	**0,828**	**0,812**	Skip	276	X	X	128	10
SVM		0,364		0,534			X	X	512	
SVM		0,203		0,582			X		512	
SVM		0,259		0,540				X	512	
SVM		0,315		0,588					512	

False Negatives

- Out of 499 rows
  479 have min 1 False Negative

- 40 of them include year numbers



False Negatives per language

fi = Finnish
vi = Vietnamese
hi = Hindi
so = Somali
ar = Arabic

	Language	False Negatives	Number of rows	False negatives per row
1	fi	1294	44	29.4
0	vi	1368	50	27.4
2	fr	1177	49	24.0
3	it	706	40	17.6
5	hi	652	48	13.6
11	so	38	3	12.7
6	ar	518	42	12.3
10	pt	61	5	12.2
4	de	656	55	11.9
7	sv	474	42	11.3
9	ca	320	41	7.8
8	en	439	62	7.1
14	ne	11	2	5.5
13	et	12	3	4.0
15	fa	4	1	4.0
12	es	29	9	3.2
16	pl	2	2	1.0
17	no	1	1	1.0

More findings

Ok ... sorry

Number are not preprocessed properly

```
     False negatives:   " 1 . 005
Actual hallucinations:   " 1 . 005
Recall: 0.0
        From response: der italienisch Gemeinde " Ponzano " liegen auf ein Höhenlage von etwa 1 . 005 Meter . Quelle : < | im _ end | "
        Query: auf welcher Höhe liegen der italienisch Gemeinde Ponzone ? .
```

Speaking of punctuation ...

Out of **7762** false negative tokens

2158 are punctuation!

Finnish and Hindi texts on average had twice as many false negative punctuations.

What is this?

An image?

Not always prepended by "Quelle:"

appears in ~20 rows

only after German or French texts

Questions?