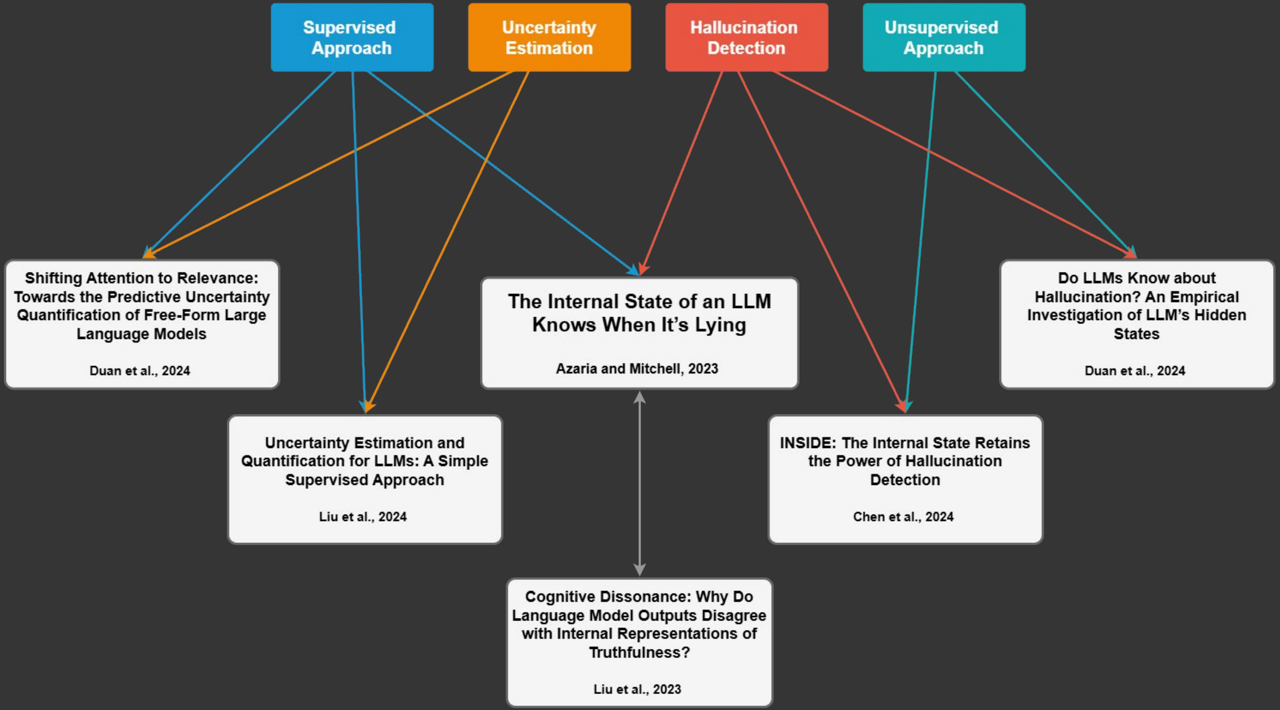
Study on Hallucination Detection by LLMs hidden state analysis

Final Presentation

Arianna Paolini - 1943164
Alessandro Scifoni - 1948810
Simone Sestito - 1937764

Advanced Machine Learning course
a.y. 2024/2025

Related Works and Context



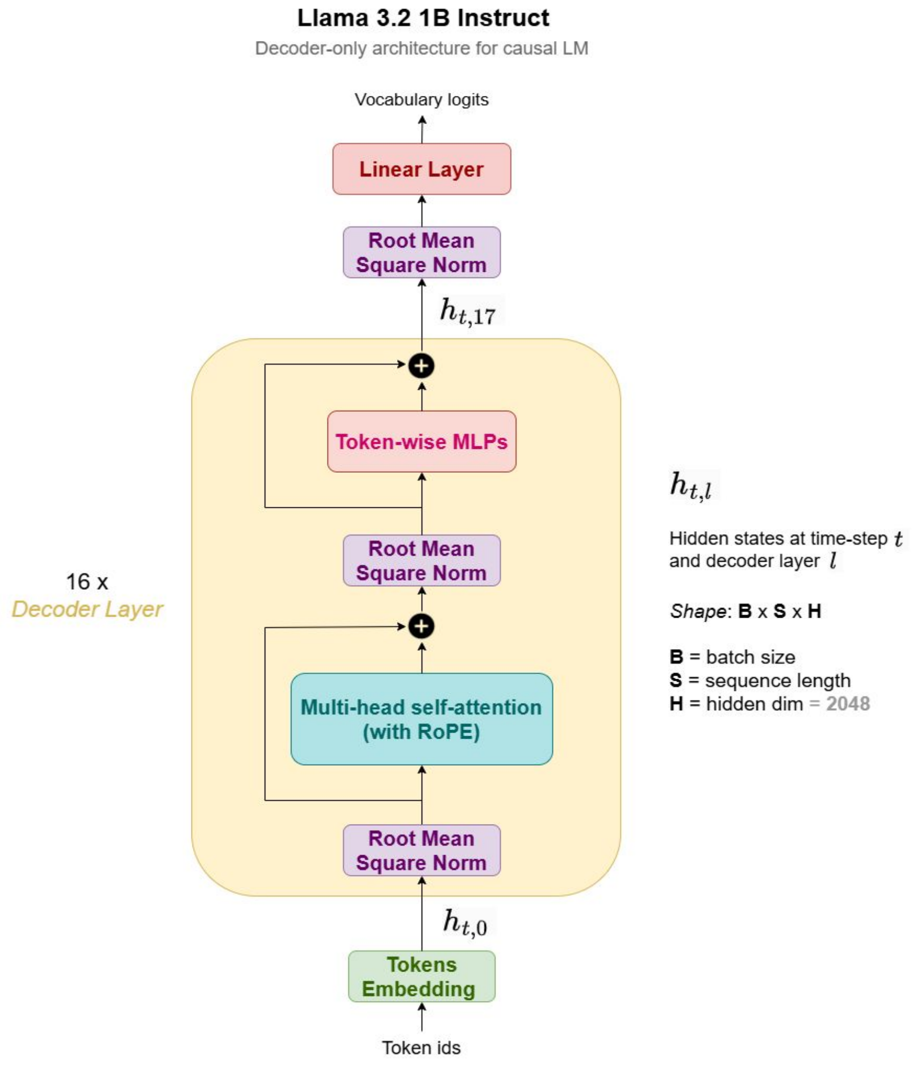
Baseline Model - SAPLMA

Starting from the original **SAPLMA** model [Azaria et al. 2023]

Recognize the **hallucinations** extracting information from **LLM hidden states**.

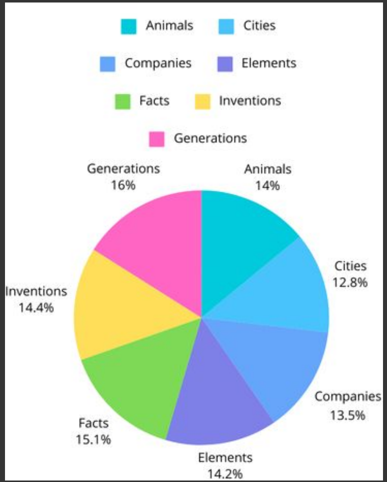
We proceeded:

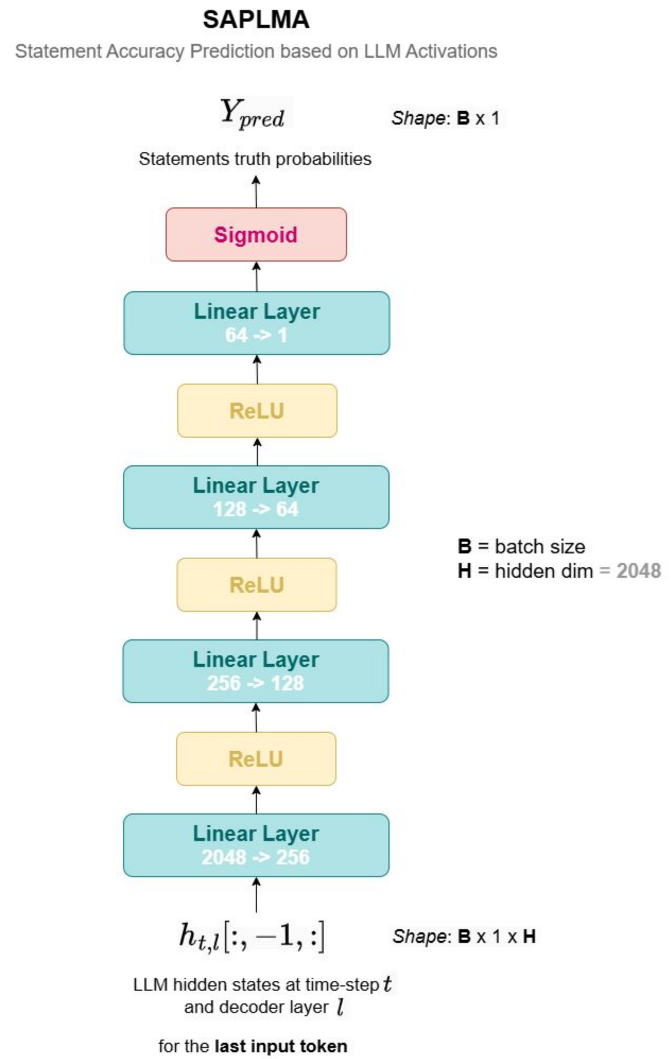
- Improving from their model with modern techniques
- Reusing their benchmarks and dataset



Baseline Model - SAPLMA Architecture

True/false dataset **6,084** sentences from **6** different topics + **245** statements generated by the *OPT 6.7b LLM* paired with a binary truth label





Baseline Model - SAPLMA Training

Technical details:
- ➔ number of epochs = 5
- ➔ optimizer = AdamW
- ➔ learning rate = 10^-5
- ➔ batch size = 64

- **train dataset**: 4868 samples
- **validation dataset**: 1217 samples
- **test dataset** (“*generated*” topic): 245 samples

The dataset was split using:
- **one topic as test set**
- the remaining as train/val

Require SAPLMA to **extract the LLM’s internal belief**, rather than learning how information must be aligned to be classified as true

Baseline Model - SAPLMA Training

Technical details:

- ➔ number of epochs = 5
- ➔ optimizer = AdamW
- ➔ learning rate = 10^-5
- ➔ batch size = 64

- **train dataset**: 4868 samples
- **validation dataset**: 1217 samples
- **test dataset** (“*generated*” topic): 245 samples

Two strategies (***reductions***) for determining the **input of SAPLMA** based on Llama hidden states:

“Last” reduction

= take the hidden states of the last input token

$$X = h_{t,l}[\ :\ , -1,\ :\ ]$$

“Mean” reduction

= average the hidden states of all input tokens at the chosen layer

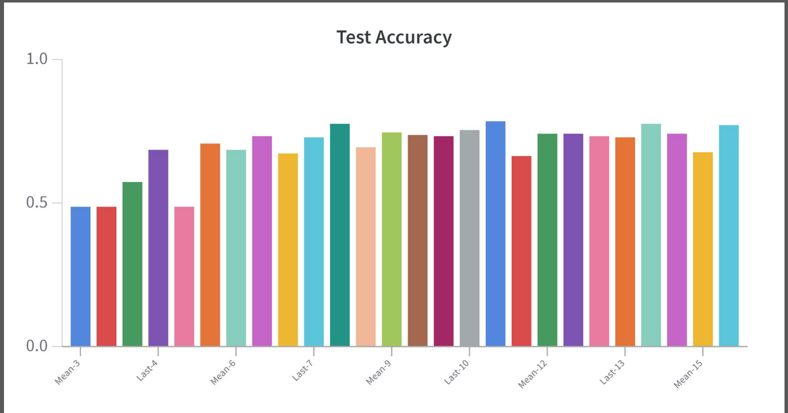
$$X = 1/S * \sum_{i=0}^{S-1} h_{t,l}[\ :\ , i,\ :\ ]$$

Baseline Model - SAPLMA Training

Technical details:

- ➔ number of epochs = 5
- ➔ optimizer = AdamW
- ➔ learning rate = 10^-5
- ➔ batch size = 64

- **train dataset**: 4868 samples
- **validation dataset**: 1217 samples
- **test dataset** (“*generated*” topic): 245 samples



We have run a sweep on *Weights & Biases* to try different combinations of:

- **hidden layer index**
- **reduction type**

and achieved a **best test accuracy of 76%**

What layer to consider? How to extract information?

Idea: hidden states from middle layers are better

What layer to consider? How to extract information?

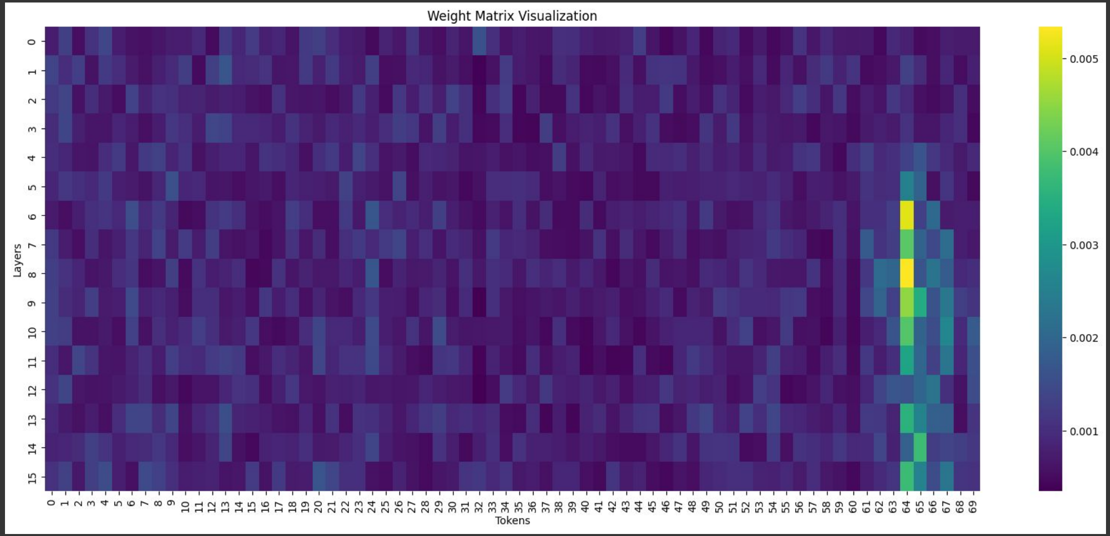
Idea: hidden states from middle layers are better

Experiment: learn the weights in the aggregation

$$w'_{i,j} = \begin{cases} w_{i,j} & if\ a_j = 1 \\ -\inf & otherwise \end{cases}$$

$$X_{learnt} = \sum_{i=0}^{L-1} \sum_{j=0}^{S-1} w'_{i,j} * h_{t,i}[:, j, :]$$

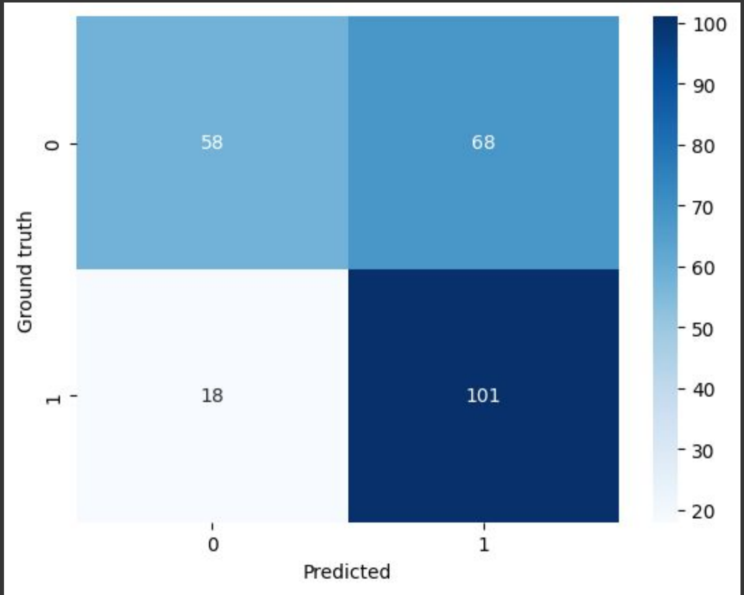
$$\sum_{i=0}^{L-1} \sum_{j=0}^{S-1} w'_{i,j} = 1$$

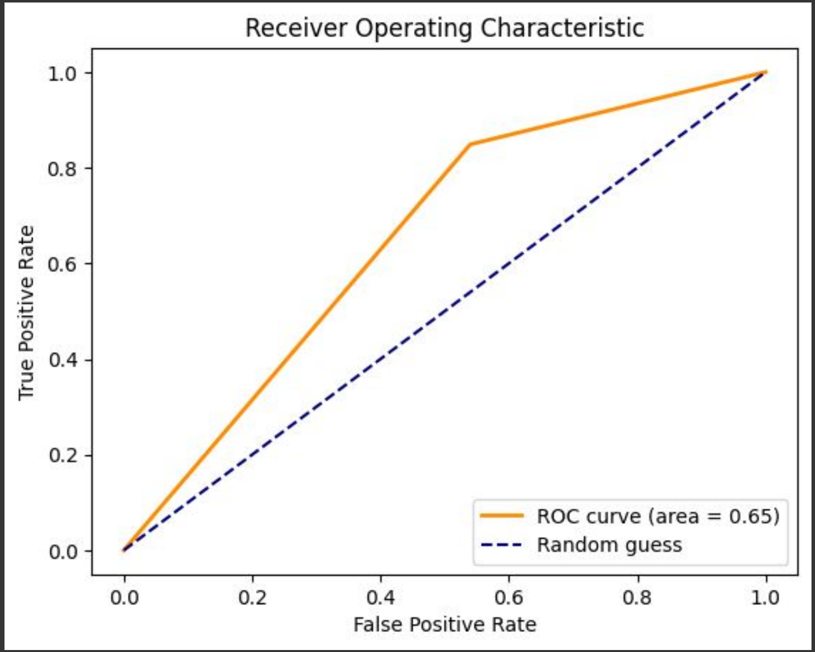


Learnable Layers Contribution: inference

```
Input: Jauba is located at the junction of the Equator and the Nile., Prediction: False
Input: Plymouth's zip code is 02360., Prediction: False
Input: Austin is the capital of the state., Prediction: True
Input: Ottawa also has a large French-speaking population., Prediction: True
Input: Lima gets an average of 1 hour of sunshine per day., Prediction: False
Input: NewDelhi is also the capital city of Delhi., Prediction: False
Input: Ashgabat is located in Turkmenistan., Prediction: True
Input: Ottawa is located in Ontario, Canada., Prediction: True
Input: Georgetown was founded in 1836., Prediction: False
Input: Road town is the capital of Virgin Islands and the largest city in the British virgin islands., Prediction: False
```

Learnable Layers Contribution: evaluation

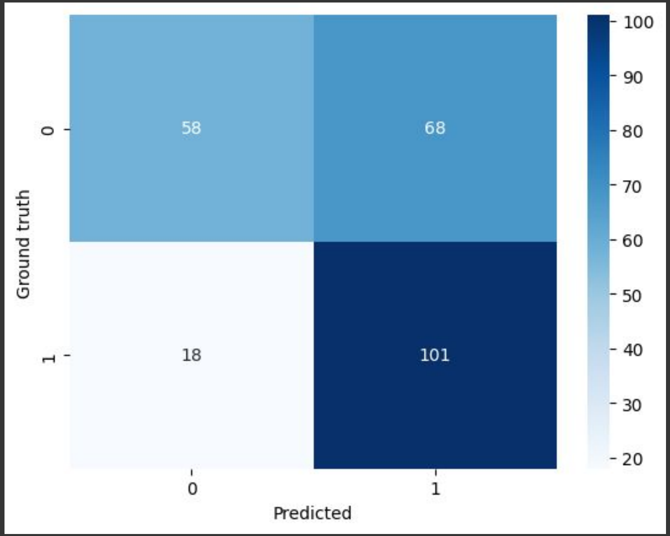




What threshold to use?

Idea: can we do better than using threshold = 0.5?

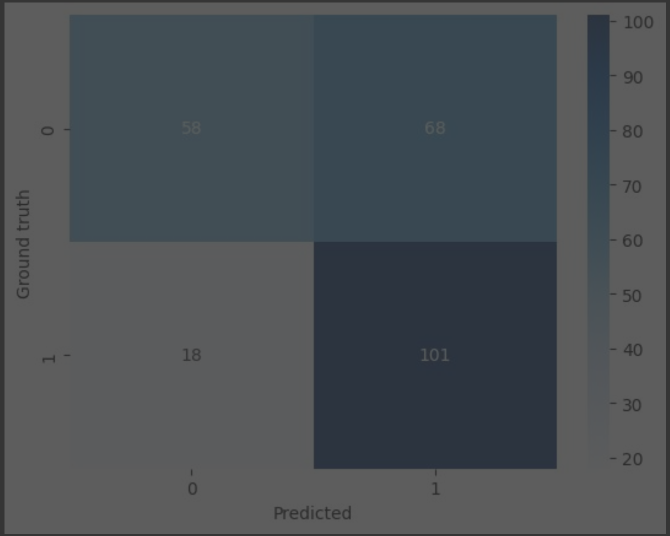
Visualize the confusion matrix

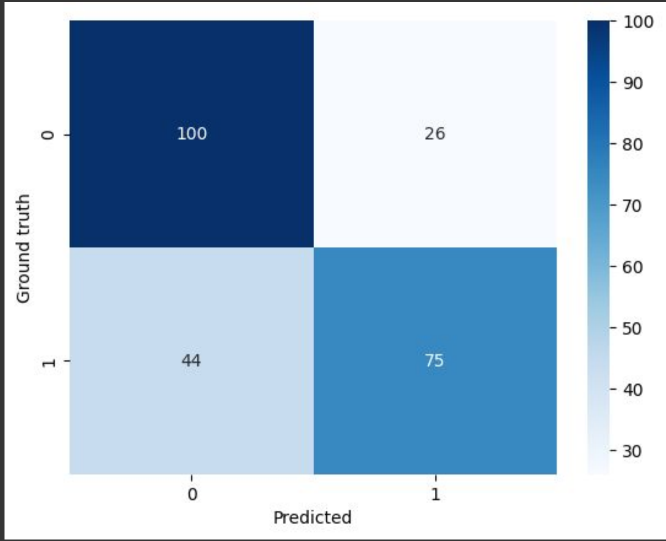


What threshold to use?

Idea: can we do better than using threshold = 0.5?

Visualize the confusion matrix





With threshold = 0.9, improving accuracy by 6%

What's the best NN architecture?

Choose the best layer, normalization, depth and width

Search best configuration, performing **few tests and keep refining**

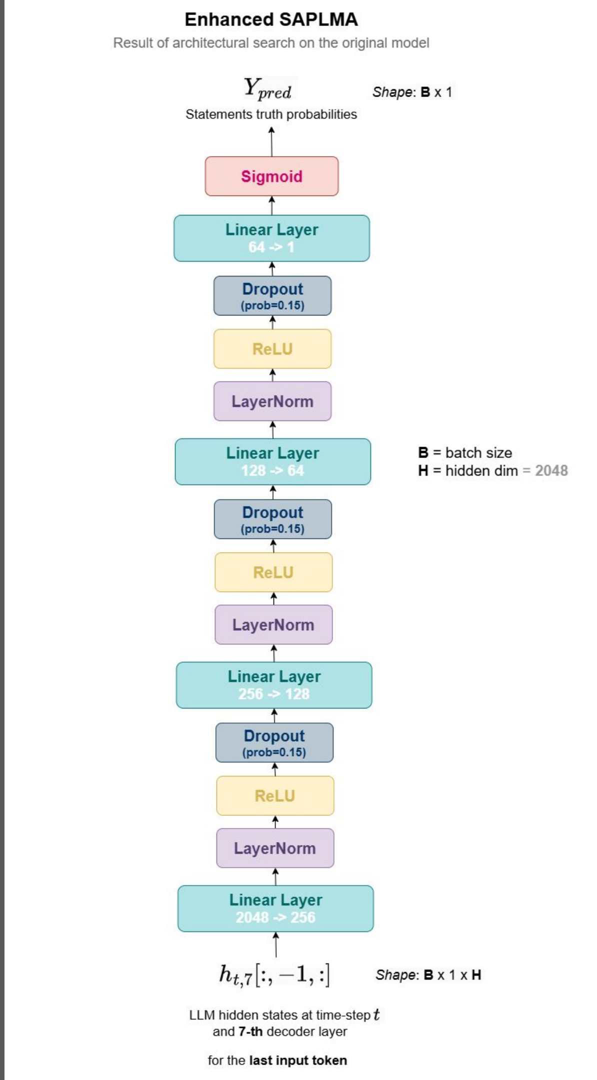
SAPLMA CONFIG	Test accuracy	Training accuracy	Validation accuracy
Layer **7**, **layer** norm, **256, 128, 64**	0.78776	0.75	0.73213
Layer **7**, **layer** norm, **256, 128, 128, 64**	0.78367	0.75	0.69597
Layer **11**, **layer** norm, **256, 64, 64, 64, 64**	0.77551	1	0.71734
Layer **11**, **batch** norm, **256, 64, 64, 64, 64**	0.76735	0.5	0.76335
Layer **15**, **batch** norm, **256, 128, 64**	0.75102	0.5	0.77239

Improved Architecture

Neural Networks can perform much better with the help of regularization

- Dropout
- LayerNorm

Hyperparameters found, performing a grid search with WandB Sweep

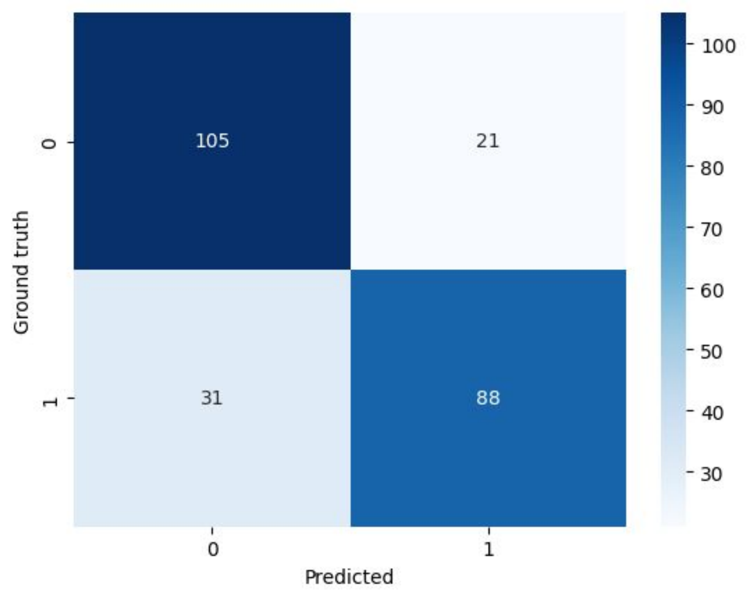


Improved Architecture

Neural Networks can perform much better with the help of regularization

- Dropout
- LayerNorm

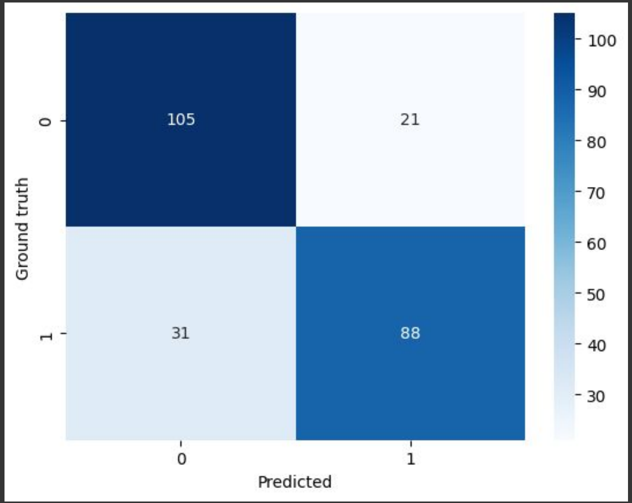
Hyperparameters found, performing a grid search with WandB Sweep

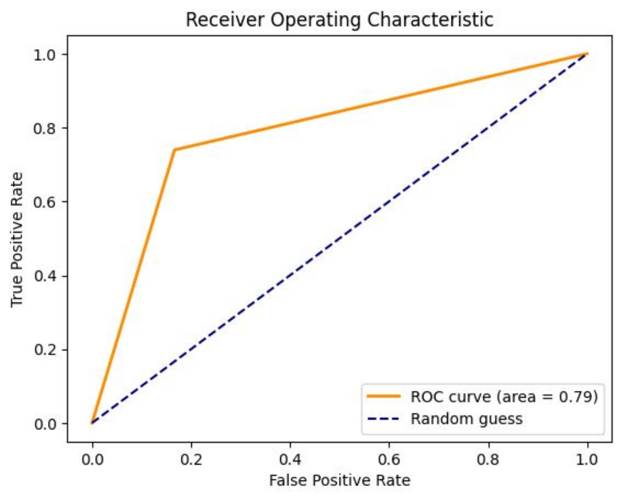


We can also use 0.5 as a threshold again

Improved Architecture: evaluation

	precision	recall	f1-score	support
0	0.77	0.83	0.80	126
1	0.81	0.74	0.77	119
accuracy			0.79	245
macro avg	0.79	0.79	0.79	245
weighted avg	0.79	0.79	0.79	245





Can it detect while generating?

Ideally, we may want to detect the hallucinations **while generating them**, and not at the end:

- infer SAPLMA on every token while generating = the last one at any given time

Understand **<u>why</u>** it detects an hallucination:

- compute the gradients on the tokens, to understand their importance in hallucination detection

```
y_true: 0 (hallucination) -- y_pred: 0.33
   SAPLMA infer on single tokens: Lima is a name of a country.
  SAPLMA gradients on embeddings: Lima is a name of a country.
```

```
y_true: 0 (hallucination) -- y_pred: 0.11
   SAPLMA infer on single tokens: Bank of China has headquarters in France.
  SAPLMA gradients on embeddings: Bank of China has headquarters in France.
```

```
y_true: 0 (hallucination) -- y_pred: 0.58
   SAPLMA infer on single tokens: The largest ocean in the world is the Indian Ocean.
  SAPLMA gradients on embeddings: The largest ocean in the world is the Indian Ocean.
```

How does SAPLMA detect hallucinations?

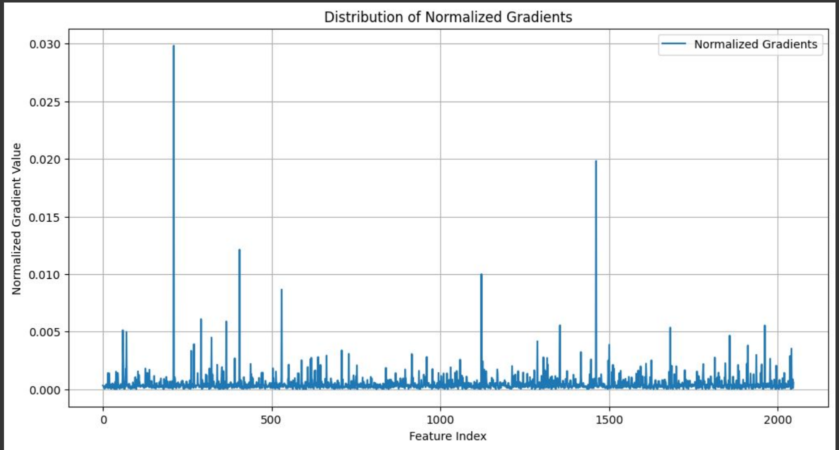
Question: is there a specific set of **LLM hidden state features** that can give us information about the truthfulness of input sentences?

Experiment: Compute gradients of SAPLMA output on a big batch of prompts with respect to the features of the LLM hidden states

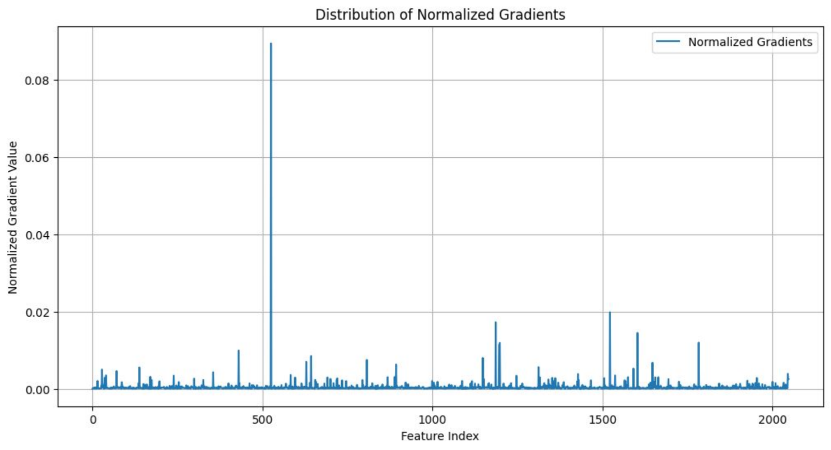
$$\nabla_b \, f(h_{0,l}) = \sum_{i=0}^{2047} \frac{\partial f}{\partial \, h_{0,l}[b, 64, i]} \, \hat{\imath}_i$$

$$\nabla \, f(h_{0,l}) = 1/B * \sum_{b=0}^{B-1} \boxed{\nabla_b \, f(h_{0,l})}$$

How does SAPLMA detect hallucinations?

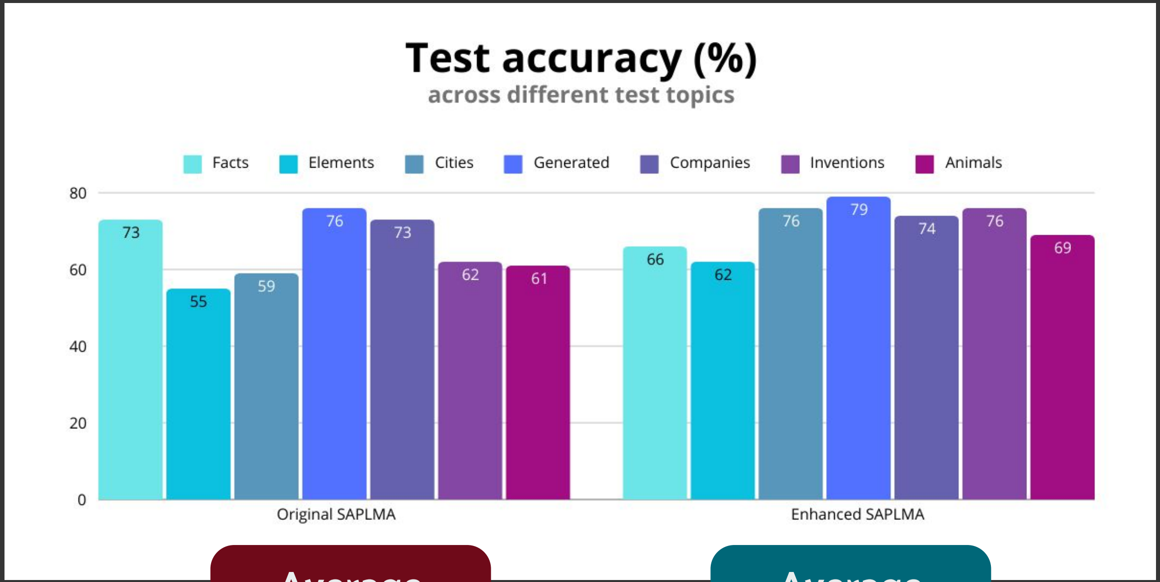


SAPLMA based on multiple hidden layers summed up with learnt weights



Enhanced SAPLMA
(based only on the 7th hidden layer)

Final Evaluation



Average
65 %

Average
72 %

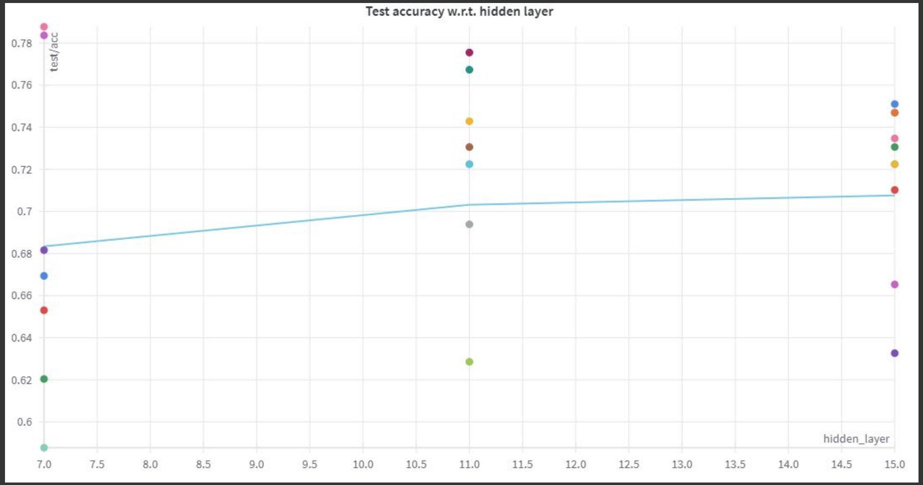
Baseline: Original SAPLMA architecture based on **12th** LLM hidden layer

Enhanced SAPLMA: architecture with layer normalization and dropout, based on **7th** LLM hidden layer

Conclusions

Lesson learnt

- **LLM hidden states** are **useful** for revealing information about **truthfulness** of statements
- **different hidden layers** encode different information
- the best performance is given by using **middle layers**



Conclusions

Limitations

- LLM hidden states are **not always available** to access
- We experimented on a **small dataset** (6k total examples)
- We experimented on a **"small" LLM** (1B parameters)
- We did **not include** all **modern training** techniques (*e.g.* learning rate decay)

Future works

- Use an **ensemble** of SAPLMA and **entropy-based** methods
- Try to increase **SAPLMA interpretability** on real-time inference

Links & References

Our code

Implemented from scratch with PyTorch Lightning

Our GitHub repository with code and experiments

Dataset from:

GitHub dataset repository

Related Works

- *The Internal State of an LLM Knows When It's Lying*, Azaria and Mitchell, 2023 (https://arxiv.org/pdf/2304.13734)
- *INSIDE: LLM's Internal State Retains the Power of Hallucination Detection*, Chen et al., 2024 (https://arxiv.org/pdf/2402.03744)
- *Do LLMs Know about Hallucination? An Empirical Investigation of LLM's Hidden States*, Duan et al., 2024 (https://arxiv.org/pdf/2402.09733)
- *Shifting Attention to Relevance: Towards the Predictive Uncertainty Quantification of Free-Form Large Language Models*, Duan et al., 2024 (https://arxiv.org/pdf/2307.01379)
- *Cognitive Dissonance: Why Do Language Model Outputs Disagree with Internal Representations of Truthfulness?*, Liu et al., 2023 (https://arxiv.org/pdf/2312.03729)
- *Uncertainty Estimation and Quantification for LLMs: A Simple Supervised Approach*, Liu et al., 2024 (https://arxiv.org/pdf/2404.15993)

Thank you!

Any questions?

GitHub repository with code and experiments:
https://github.com/simonesestito/AML-project