IPDF2LATEX

$$\det\left(\frac{\partial^2 F}{\partial x_i \partial x_j}\right) = \sum_{\sigma \in S_n} sgn(\sigma)$$

```
\det\left( \frac{\partial^2 F}{\partial x_i \partial x_j}
\right) = \sum_{\sigma \in S_n} \text{sgn}(\sigma)
```

Macéo Ottavy, **Mathieu Longatte**, **Louison Mocq**, **Ankit Gayen**

Supervised by **Simon Delamare**

and the rule is proved that

$$\frac{du^s}{dx} = nu^{s-1}\frac{du}{dx},$$

where n is a positive fraction whose numerator and denominator are integers. This rule has already been used in the solution of numerous exercises.

34. The Derivative of a Constant. Let $y = c$, where c is a constant. Corresponding to any Δx, $\Delta y = 0$, and consequently

$$\frac{\Delta y}{\Delta x} = 0,$$

and

$$\lim_{\Delta x \to 0} \frac{\Delta y}{\Delta x} = 0,$$

or

$$\frac{dy}{dx} = 0.$$

The derivative of a constant is zero.

Interpret this result geometrically.

35. The Derivative of the Sum of Two Functions. Let

$$y = u + v,$$

where u and v are functions of x. Let Δu, Δv, and Δy be the increments of u, v, and y, respectively, corresponding to the increment Δx.

$$y + \Delta y = u + \Delta u + v + \Delta v$$
$$\Delta y = \Delta u + \Delta v$$
$$\frac{\Delta y}{\Delta x} = \frac{\Delta u}{\Delta x} + \frac{\Delta v}{\Delta x}$$
$$\frac{dy}{dx} = \frac{du}{dx} + \frac{dv}{dx},$$

or

$$\frac{d(u+v)}{dx} = \frac{du}{dx} + \frac{dv}{dx}.$$

The derivative of the sum of two functions is equal to the sum of their derivatives.

and the rule is proved that

$$\frac{du^s}{dx} = nu^{s-1}\frac{du}{dx},$$

where n is a positive fraction whose numerator and denominator are integers. This rule has already been used in the solution of numerous exercises.

34 The Derivative of a Constant

Let $y = c$, where c is a constant. Corresponding to any Dx, $Dy = 0$, and consequently

$$\frac{\Delta y}{\Delta x} = 0,$$

and

$$\lim_{\Delta x \to 0} \frac{\Delta y}{\Delta x} = 0,$$

or

$$\frac{dy}{dx} = 0.$$

The derivative of a constant is zero.

Interpret this result geometrically.

35 The Derivative of the Sum of Two Functions

Let

$$y = u + v,$$

where u and v are functions of x. Let Du, Du, and Dy be the increments of u, v, and y, respectively, corresponding to the increment Dx.

$$y + \Delta y = u + \Delta u + v + \Delta v$$

$$\Delta y = \Delta u + \Delta v$$

$$\frac{\Delta y}{\Delta x} = \frac{\Delta u}{\Delta x} + \frac{\Delta v}{\Delta x}$$

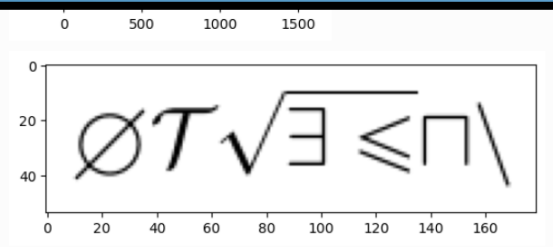
$$\frac{dy}{dx} = \frac{du}{dx} + \frac{dv}{dx},$$

or

$$\frac{d(u+v)}{dx} = \frac{du}{dx} + \frac{dv}{dx}.$$

The derivative of the sum of two functions is equal to the sum of their derivatives.

Website




```
<class 'list'>
```

Creating dataset

We finally want to create a dataset wich will be a map containing two fields.

```
[8]: ds = {"images":[], "latex-formula":[]}
```

Then we add all the images and corresponding latex-formula we want.

```
[9]: ds["images"].append(img)
     ds["latex-formula"].append(formula)
```

Finally we save our dataset thanks to the `json` package.

```
[10]: import json

      json_object = json.dumps(ds)

      ds_name = "files/image-latex-datasets.json"
      with open(ds_name, "w") as f :
          f.write(json_object)
```

To open your dataset, do this:

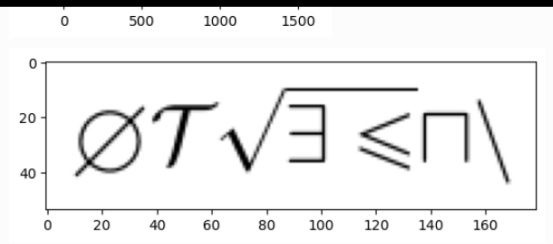
```
[11]: with open(ds_name, 'r') as f:
          data = json.load(f)

      print(data["latex-formula"])
```

```
[' \\emptyset \\mathcal{ T \\sqrt{ 9 \\leq } u n }']
```

latest

Website




```
<class 'list'>
```

Creating dataset

We finally want to create a dataset wich will be a map containing two fields.

```
[8]: ds = {"images":[], "latex-formula":[]}
```

Then we add all the images and corresponding latex-formula we want.

```
[9]: ds["images"].append(img)
     ds["latex-formula"].append(formula)
```

Finally we save our dataset thanks to the `json` package.

```
[10]: import json

      json_object = json.dumps(ds)

      ds_name = "files/image-latex-datasets.json"
      with open(ds_name, "w") as f :
          f.write(json_object)
```

To open your dataset, do this:

```
[11]: with open(ds_name, 'r') as f:
          data = json.load(f)

      print(data["latex-formula"])
```

```
[' \\emptyset \\mathcal{ T \\sqrt{ 9 \\leq } u n }']
```

GitHub

GeckSpy / IPdf2Latex

Code | Issues | Pull requests | Actions | Projects | Wiki | Security | Insights | Settings

IPdf2Latex Public — Pin | Unwatch 1 | Fork 0 | Star 0

main | 1 Branch | 0 Tags | Go to file | Add file | Code

ankitgayen2021 Update metrics.ipynb — 2d655c6 · 17 hours ago — 191 Commits

Presentation	clean up	2 days ago
assets	update	yesterday
notebook	Update metrics.ipynb	17 hours ago
.DS_Store	changes	3 months ago
CONTRIBUTING.md	Create CONTRIBUTING.md	yesterday
README.md	update	yesterday
TODO.md	update	yesterday
tutorials-idea.md	Update tutorials-idea.md	last month

About

No description, website, or topics provided.

Readme
Activity
0 stars
1 watching
0 forks

Releases

No releases published
Create a new release

Packages

No packages published
Publish your first package

Contributors 4

GeckSpy
mathieulngt
ankitgayen2021 Ankit Gayen
loumock

README

IPdf2Latex

IPdf2LaTeX is a student project aimed at creating in-depth educational resources to help anyone with Python knowledge learn how to develop AI models that convert images into their LaTeX code equivalents.

We have built a website (Tutorial IA) containing a series of notebooks that cover all the essential concepts you need.

Objective: Image to Latex

$$\det\left(\frac{\partial^2 F}{\partial x_i \partial x_j}\right) = \sum_{\sigma \in S_n} sgn(\sigma) \prod_{i=1}^{n} \frac{\partial^2 F}{\partial x_i \partial x_{\sigma(i)}}$$

```
\det\left( \frac{\partial^2 F}{\partial x_i \partial x_j} \right)
    = \sum_{\sigma \in S_n} \text{sgn}(\sigma) \prod_{i=1}^n
    \frac{\partial^2 F}{\partial x_i \partial x_{\sigma(i)}}
```

Steps

Step 1: Collect data

Easy: Generate dataset from scratch or arxiv
or take one from Huggingface

Related tutorials: Basics, Images, Loading , Building

Steps

Step 1: Collect data

Easy: Generate dataset from scratch or arxiv
or take one from Huggingface

Related tutorials: Basics, Images, Loading , Building

Step 2: Chose a relevant architecture

Hard: No obvious structure for Image to text
Hard: Language models require a gigantic amount of data

Related tutorials: Curse of Dimensionality, Summary of the tokenizers, Fine-tune a pretrained model

Transformers

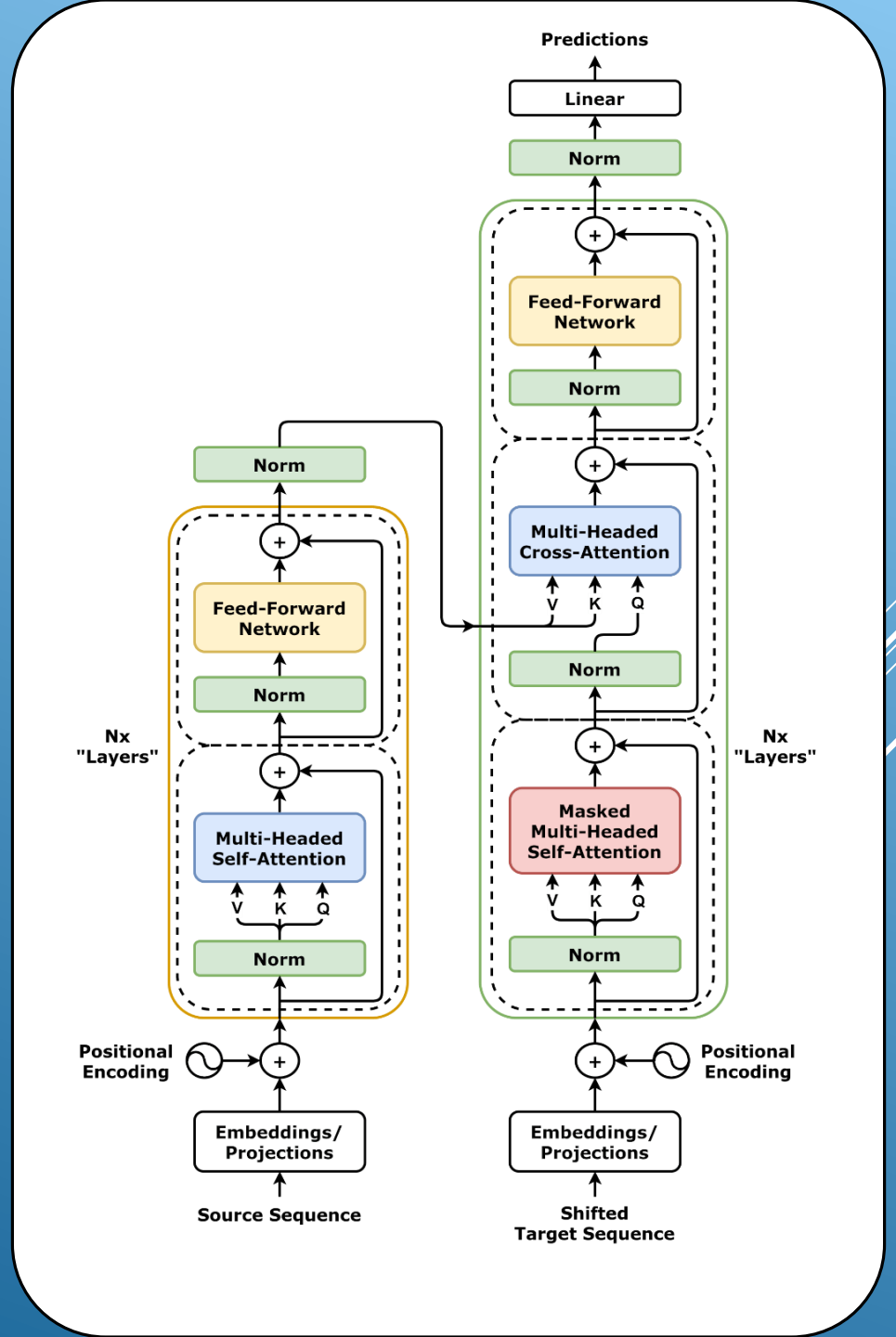
Objectives :

Capture long distance dependency

Enable parallelization in the treatment

Time :	Yesterday,
Person :	Ryan
Action :	bought
Object :	a red car
Goal :	to get to his job
Place :	in Paris

Transformers

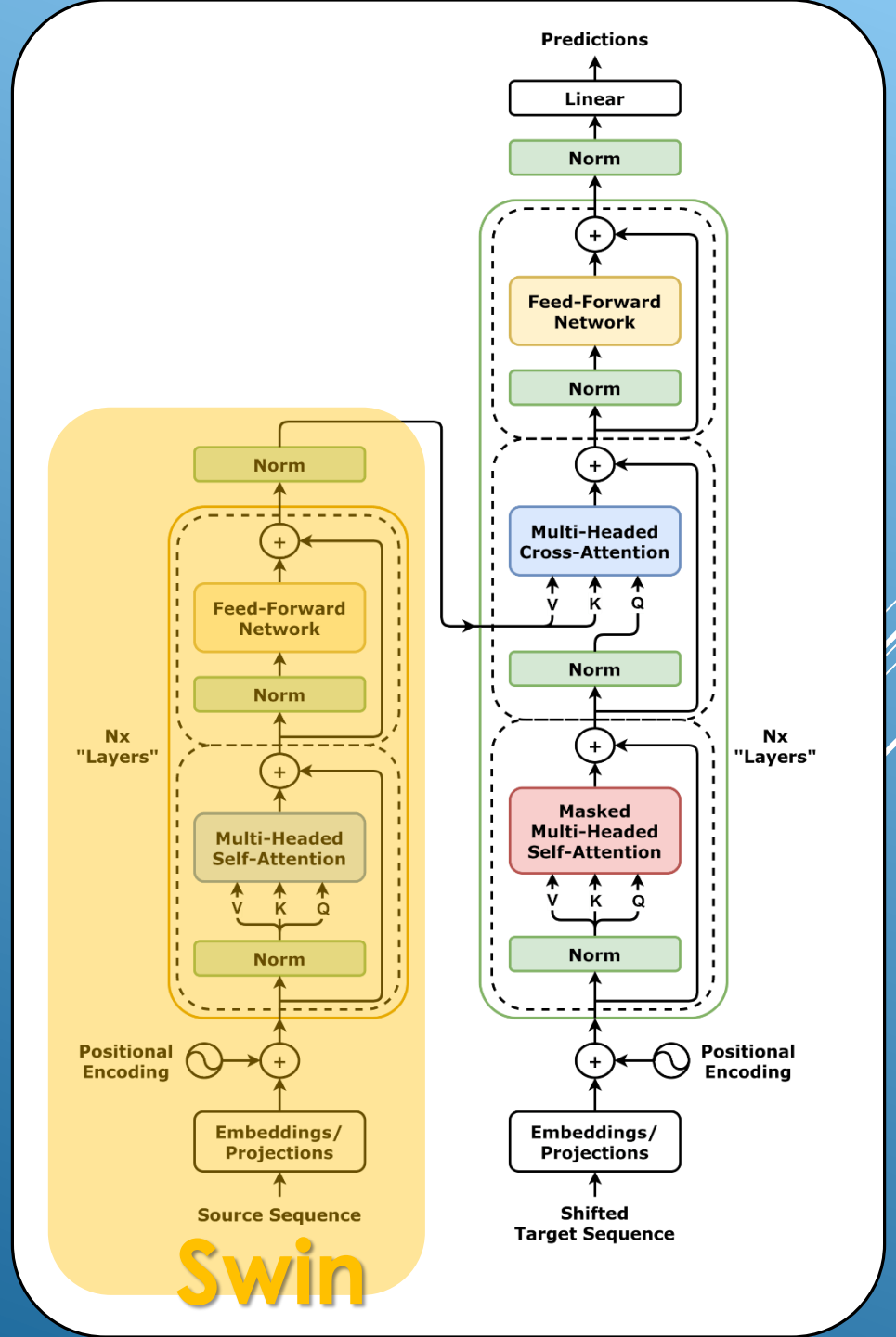


Related tutorials: Transformers

Transformers

Encoder :

An encoder in AI transforms input data (e.g., text, images) into a compact numerical representation that captures essential features.



Related tutorials: Transformers

Transformers

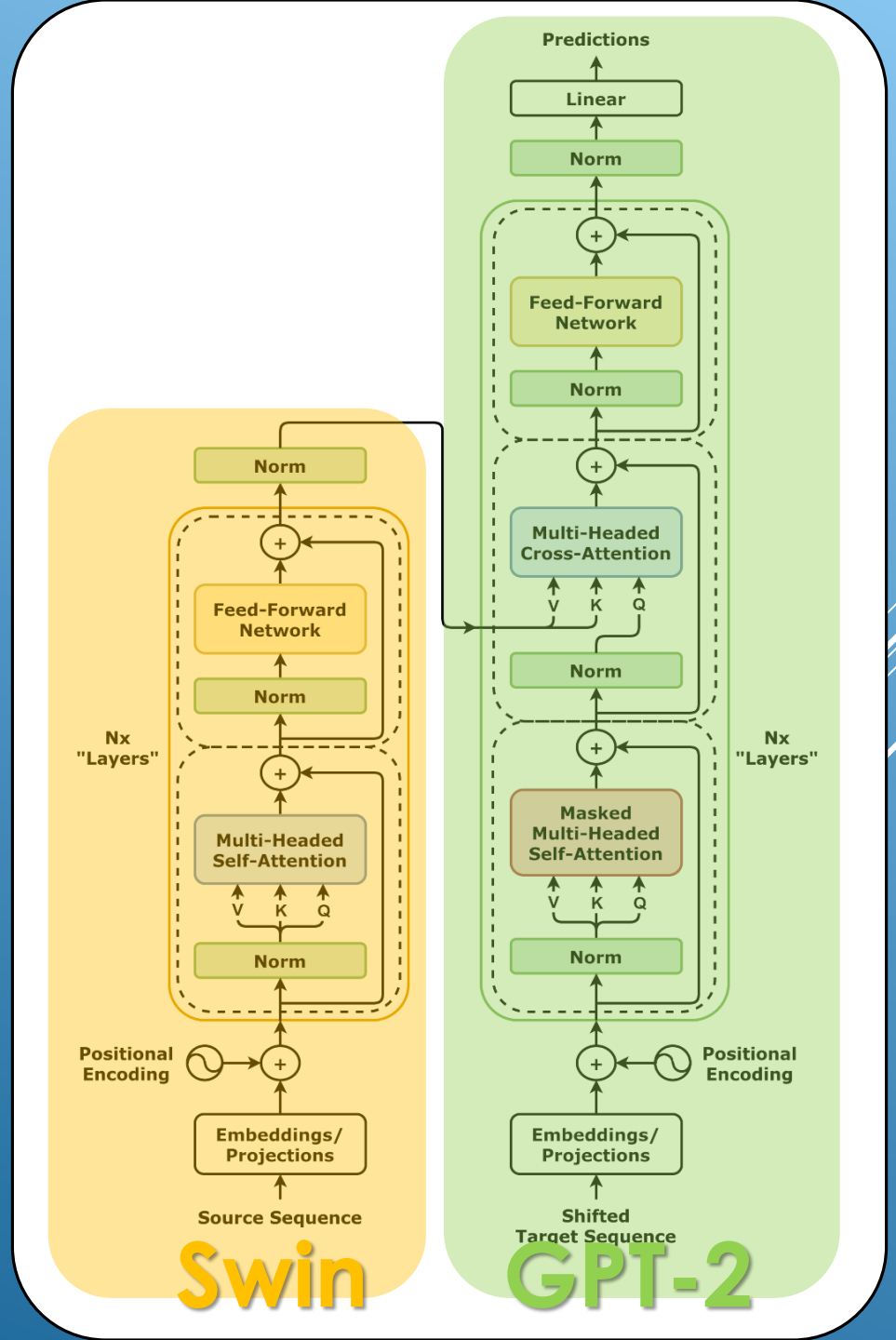
Encoder :

An encoder in AI transforms input data (e.g., text, images) into a compact numerical representation that captures essential features.

Decoder :

A decoder in AI takes this encoded representation and reconstructs it back into a human-interpretable format, such as text or images.

Related tutorials: Transformers

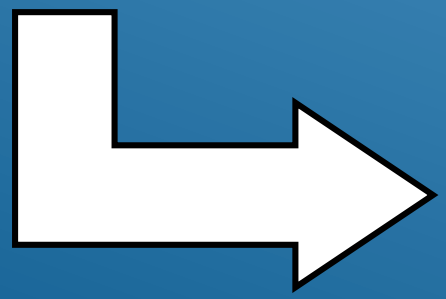


Steps

Step 1: Collect data

Step 2: Chose a relevant architecture

Step 3: Chose metrics to evaluate the model



BLEU metric

Related tutorials: Introduction to few NLP metrics

BLEU metric

BLEU = BiLingual Evaluation Understudy

We are going to predict mathematical formula

BLEU metric

BLEU = BiLingual Evaluation Understudy

```
We are going to predict mathematical formula
```

Is it good?

Goal of a metric:

Evaluate efficiently the performances of the model

BLEU metric

N-gram = sequence of n consecutive words

```
We are going to predict mathematical formula
```

BLEU metric

N-gram = sequence of n consecutive words

We are going to predict mathematical formula

1-grams: 

BLEU metric

N-gram = sequence of n consecutive words

We are going to predict mathematical formula

1-grams:

2-grams:

BLEU metric

Modified n-grams:

$$\frac{\sum_{C \in unique\ n-grams} count\_clip(C)}{number\ of\ n-grams}$$

BLEU metric

Modified n-grams:

$$\frac{\sum_{\mathcal{C} \in unique\ n-grams} count\_clip(\mathcal{C})}{number\ of\ n-grams}$$

n-gram precision:

$$p_n = \frac{\sum_{\mathcal{C} \in \{Candidates\}} \sum_{n-gram \in \mathcal{C}} count\_clip(\text{n-gram})}{\sum_{\mathcal{C}' \in \{Candidates\}} \sum_{n-gram \in \mathcal{C}'} count(\text{n-gram})}$$

BLEU metric

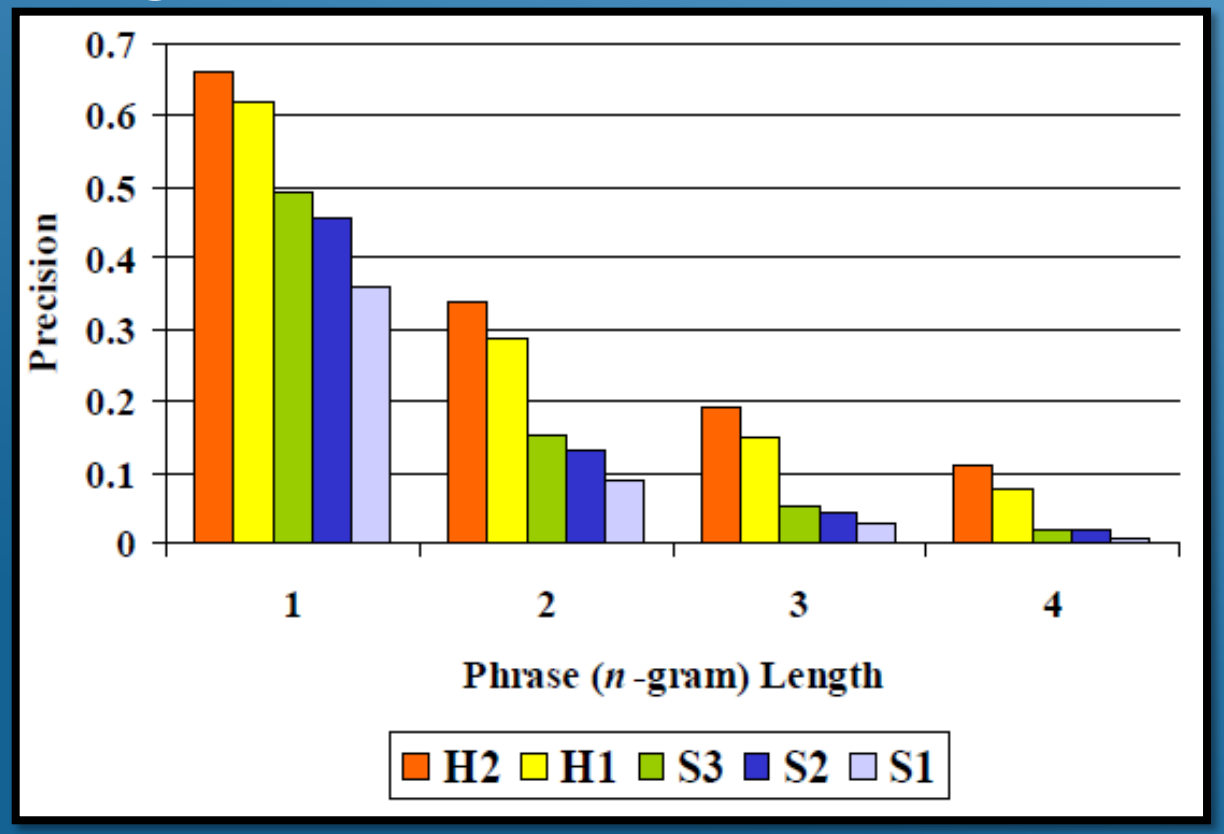
Low n: Checks the vocabulary of tokens/words used in the candidate

High n: Checks and ensures proper token order and longer syntactic rules

BLEU metric

Low n: Checks the vocabulary of tokens/words used in the candidate

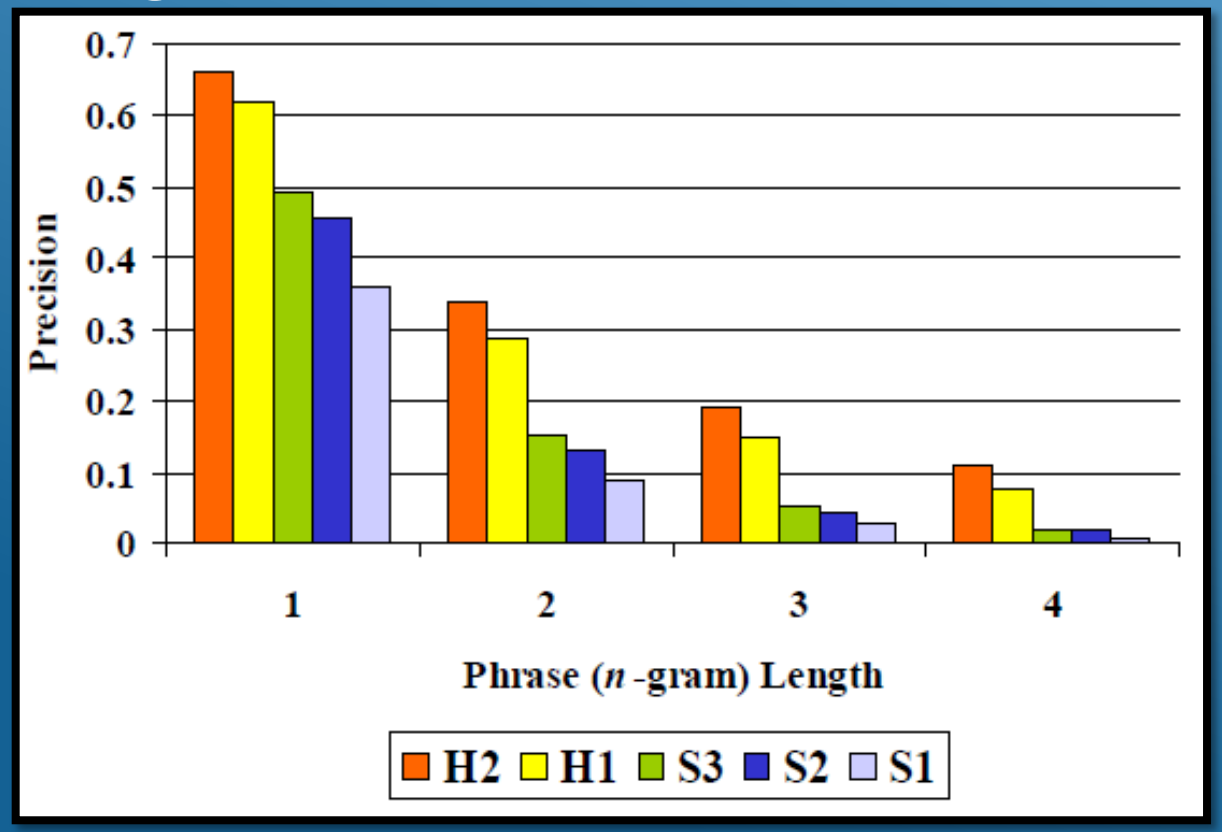
High n: Checks and ensures proper token order and longer syntactic rules



BLEU metric

Low n: Checks the vocabulary of tokens/words used in the candidate

High n: Checks and ensures proper token order and longer syntactic rules



$$BLEU = BP.\exp\left(\sum_{n=1}^{N} w_n \log p_n\right)$$

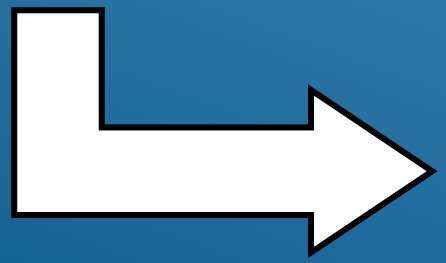
Steps

Step 1: Collect data

Step 2: Chose a relevant architecture

Step 3: Chose metrics to evaluate the model

Step 4: Chose a relevant programming environment



High view library by **Huggingface**: *Transformers, Trainer*
HPC platform: grid5000

Related tutorials: finetunin a pre-existing model, Trainer-a PyTorch optimized training loop

Experiments and Results

Training:

- 30 hours on 4 GPUs on grid5000
- We finetuned 600K paraters among 250M parameters
- 250K examples on the training dataset, during 3 epochs

Results:

Model	Google BLEU	Data Size	Parameters
Im2Latex	*0.67*	441K	243M
TexTeller	*0.77*	7.5M	300M
Pix2Text	*0.07*	100K	25M
Sumen	*0.47*	6.9M	350M
Our	**0.23**	**250K**	**250M**

Experiments and Results

Before training

$$\Delta_+(x,y) = \sum_{n=-\infty}^{\infty} \tilde{\Delta}(x_0 + nT, \vec{x}; y_0, \vec{y})$$

I don't know if it's a coincidence or not," he said. "I think it's just a matter of time before we get to know each other."

He added: "It's not like we're going to be friends. It's just that we're not going to get along very well.

"We're not friends. We're not even friends. I don't think we'll ever get along. I think we're just going to have to get used to each other. We'll just have to figure out how to get through this together. This article is from the archive of our partner .

Experiments and Results

After training

$$\Delta_+(x,y) = \sum_{n=-\infty}^{\infty} \tilde{\Delta}(x_0 + nT, \vec{x}; y_0, \vec{y})$$

```
{ \Delta } _ { + } ( x , y ) = \sum _ { n = - \infty } ^ { { } } { \tilde {
\Delta \left( } x _ { 0 } + n T , { \vec { x } } ; y _ { 3 } ;
{ \bf { y } } \right) } . \ , | {% 1 2 3 5 6 7 8 9 0 4 ^ ( ) x p = y d e g f a
b c n t v q z ~ r s o i u w " j l k > / V _ I + J - : } \q A
```

Experiments and Results

After training

$$\Delta_+(x,y) = \sum_{n=-\infty}^{\infty} \tilde{\Delta}(x_0 + nT, \vec{x}; y_0, \vec{y})$$

```
{ \Delta } _ { + } ( x , y ) = \sum _ { n = - \infty } ^ { { } } { \tilde {
\Delta \left( } x _ { 0 } + n T , { \vec { x } } ; y _ { 3 } ;
{ \bf { y } } \right) } . \ , | {% 1 2 3 5 6 7 8 9 0 4 ^ ( ) x p = y d e g f a
b c n t v q z ~ r s o i u w " j l k > / V _ I + J - : } \q A
```

Conlusion

- Model generalizing well
  Limitations: Training times, computation power

- Around 30 notebooks covering all project

Conlusion

- Model generalizing well
  Limitations: Training times, computation power

- Around 30 notebooks covering all project

Futur

- Improve model performance
- extend to the PDF to Latex tool
- provide the associated notebooks

IPDF2LATEX

Website: *https://tutorial-ia-pe.readthedocs.io/en/latest/*

GitHub: *https://github.com/GeckSpy/IPdf2Latex*

Macéo Ottavy, Mathieu Longatte, Louison Mocq, Ankit Gayen

Supervised by Simon Delamare

APPENDIX

References

Image-to-LaTeX Converter for Mathematical Formulas and Text

Daniil Gurgurov[1] Aleksey Morshnev[2]

[1]Department of Language Science and Technology, Saarland University

[2]Department of Data Science and Artificial Intelligence, Saarland University

{dagu00001, almo00008}@stud.uni-saarland.de

Experiments and Results

After training

$$N = \frac{1}{4\beta \left(\frac{2}{\sqrt{d_\perp}}\right)^{d_\perp}} \int_{y_N}^{y_{end}} \frac{(1-y)^{d_\perp - 1} dy}{y^3 \left(1 + (\frac{d_\perp}{4} - \frac{3}{2})y\right)}$$

```
N = \frac { 1 } { \beta \Lambda ^ { d - 1 } } \int _ { \mu _ { N } } ^ { \nu _ { 0 }
\alpha _ { s } } { ( \frac 2 { \sqrt { d _ { D } } } ) ^ { 2 } \over { y ^ { 3 } \left(
1 + ( 1 + \frac 3 { 4 } - \frac 1 4 ) y \right) } } \ .%: = , ( ) 1 2 - 3 4 5 7 8 9 6 |
> { ~ ⁄ 0 ^ e p a b c d x r v l s i y h t o k q z n I g f }
```