Language Modelling with Pixels

Phillip Rust¹  Jonas F. Lotz¹  Emanuele Bugliarello¹  Elizabeth Salesky³  Miryam de Lhoneux¹,²,⁴  Desmond Elliott¹

¹University of Copenhagen  ²KU Leuven  ³Johns Hopkins University  ⁴Uppsala University
Summary

We train a pixel-based encoder of language (PIXEL), a language model trained solely on images of rendered text.

Some of PIXEL’s strengths are

Out-of-the-box transfer to unseen languages and scripts

Robustness to orthographic attacks & code-switching*

* See our paper for the code-switching results
NLP in the Era of Scale

Emergent Abilities of Large Language Models

(Wei+ TMLR’22)

Source: www.economist.com/interactive/briefing/2022/06/11/huge-foundation-models-are-turbo-charging-ai-progress
NLP for all written languages?

There are \(~7000\) spoken languages, of which \(~3000\) are written and at least \(400\) have \(^{>1M}\) speakers.

Most NLP only covers \(100\) languages \((\text{van Esch}+ \text{LREC’22})\).

→ Lack of technological inclusion for billions of people.
What’s left? NLP for all written languages

There are ~7000 spoken languages, of which ~3000 are written and at least 400 have >1M speakers

Most NLP only covers 100 languages (van Esch+ LREC’22)
→ Lack of technological inclusion for billions of people
Question: What’s stopping us?

NLP is an open vocabulary problem.

A language model’s ability to process unseen words is determined by its vocabulary:

1. “Trained” over a corpus: Byte-Pair Encoding (Sennrich+ ACL’16) → Unseen tokens not in the vocabulary (unless w/ byte-level fallback)

2. Corpus-independent: characters (Clark+ TACL’22) / bytes (Xue+ ACL’22) → Need to deal with longer sequence lengths
Answer: The Vocabulary Bottleneck

Language models have **discrete** input and output **vocabularies** expressed over a **finite inventory** of tokens, characters, words, sub-words, etc.

→ **This creates a bottleneck in two places**

**Computational bottleneck** in the output layer

**Representational bottleneck** in the embedding layer

Vaswani+ NeurIPS'17
TL;DR of our paper

We attempt to crack the *vocabulary bottleneck* with pixels.

But what does that mean?
The NLP pipeline simplified
My cat, Dr. Beans II., sleeps 22h a day.
The NLP pipeline simplified

My cat, Dr. Beans II., sleeps 22h a day.
The NLP pipeline simplified
The NLP pipeline simplified

Raw Text → Normalization → Tokenization → Embedding Lookup → Model

<table>
<thead>
<tr>
<th>Raw Text</th>
<th>Normalization</th>
<th>Tokenization</th>
<th>Embedding Lookup</th>
<th>Model</th>
</tr>
</thead>
<tbody>
<tr>
<td></td>
<td>4.4960e-01</td>
<td>-8.0239e-02</td>
<td>-5.0748e-01</td>
<td>-8.9100e-02</td>
</tr>
<tr>
<td></td>
<td>-2.0737e-01</td>
<td>3.5708e-01</td>
<td>9.4414e-01</td>
<td>4.0695e-01</td>
</tr>
<tr>
<td></td>
<td>...</td>
<td>...</td>
<td>...</td>
<td>...</td>
</tr>
<tr>
<td></td>
<td>-9.5125e-02</td>
<td>1.1732e-01</td>
<td>1.2228e-01</td>
<td>6.2151e-01</td>
</tr>
<tr>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
</tbody>
</table>
The NLP pipeline simplified

1. **Raw Text**
2. **Normalization**
   - 6.8479e-01
   - 7.4406e-02
   - -5.7043e-02
   - ...
   - -5.6763e-02
3. **Tokenization**
   - 5.5677e-01
   - -3.8491e-01
   - 5.5702e-01
   - ...
   - 2.2008e-01
4. **Embedding Lookup**
   - 5.0547e-01
   - 9.9931e-02
   - 1.7320e-01
   - ...
   - -1.8281e-01

5. **Model**
   - 1.7858e+00
   - 2.3194e-01
   - 1.4286e-01
   - ...
   - -1.3138e-01
Cracking the *vocabulary bottleneck* with pixels

Treat *language processing* as *visual processing*
Cracking the *vocabulary bottleneck* with pixels

Treat **language processing** as **visual processing**

1. Raw Text
2. Render Text as Image
3. Model
Cracking the *vocabulary bottleneck* with pixels

Treat **language processing** as **visual processing**

My cat, Dr. Beans II., sleeps 22h a day.
Cracking the *vocabulary bottleneck* with pixels

Treat *language processing* as *visual processing*

My cat, Dr. Beans II., sleeps 22h a day.
Cracking the vocabulary bottleneck with pixels

Treat *language processing* as *visual processing*

Raw Text → Render Text as Image → Model

<table>
<thead>
<tr>
<th>Raw Text</th>
<th>Render Text as Image</th>
<th>Model</th>
</tr>
</thead>
<tbody>
<tr>
<td>-4.1020e-01 2.7750e-01 3.3202e-01 ... 2.9194e-01</td>
<td>-3.6107e-01 2.0695e-01 1.7878e+00 ... 9.4824e-02</td>
<td>3.2538e-01 1.2356e+00 -1.0839e+00 ... 6.8341e-01</td>
</tr>
<tr>
<td></td>
<td></td>
<td>1.7513e-01 1.1834e+00 -4.8054e-01 ... 6.8465e-01</td>
</tr>
</tbody>
</table>
Inspiration

Robust Open-Vocabulary Translation from Visual Text Representations
*(Salesky+ EMNLP'21)*

Masked Autoencoders are Scalable Visual Learners
*(He+ CVPR'22)*
Pixel-based **Encoder of Language (PIXEL)**

- **16x16 patch resolution**
- **Google Noto Fonts**
- **PyGame / PangoCairo**

---

**1. Render Text**

My cat 🐱 enjoys eating warm oatmeal for lunch and dinner.

**2. Projection + Position Embedding**

**3. CLS Embedding & Span Mask** $m$ patches

**MSE =** \[ \frac{1}{m} \frac{1}{n} \sum_{i=1}^{m} \sum_{j=1}^{n} (Y_{ij} - \hat{Y}_{ij})^2 \]

---

Decoder

Encoder

8 Layers

12 Layers
PIXEL learns to reconstruct text

Penguins are designed to be streamlined and hydrodynamic, so having short legs would add extra aerodynamics. Adding short legs with webbed feet to act like runggs, helps to give them that dog-like figure you can’t compare bird anatomy with humans, we would see something as peculiar. By taking a look at the side-by-side image in Figure 1, you can see how their leg bones oppose to ours. What most people mistake, penguins are actually the avian birds. This gives an illusion that bird knees bend opposite of ours. The knees are actually tucked up inside the body of the bird. So how does this look inside the penguin? In the images below, you can see boxes surrounding the penguins’ knees.

100k steps

Demo: https://huggingface.co/Team-PIXEL/pixel-base
Downstream Task Fine-Tuning

1. Render Text
   My cat ċa sits in a beautiful box full of black beans.

2. Projection + Position Embedding
   My cool cat ċa sits in a beautiful box full of black beans.

3. CLS Embedding
   My cat ċa enjoys eating warm oatmeal for lunch and dinner.
# Flexible Text Renderer

## Color Emoji

| My cat 🐱 loves pancakes 🥞 and grapes 🍇. |

## Left-to-right, right-to-left, and logosyllabic writing systems

| 一隻貓正在吃碗中的貓糧 |

| قت جاثم على غصن شجرة |

## Word-level rendering

| ሆን ይኋ ይ_MAILBOX:: ይጠን ይጠን ይጠን ይጠን ይጠን ይጠን ይጠን ይጠን ይጠን ይጠን ይጠን ይጠን ይጠን ይጠን ይጠን ይጠን ይጠን ይጠን ይጠን ይጠን ይጠን ይጠን ይጠን ይጠን ይጠን ይጠን ይጠን ይጠን ይጠን ይጠን ይጠን ይጠን ይጠን ይጠን ይጠን ይጠን ይጠን ይጠን ይጠን ይጠን ይጠን ይጠን ይጠን ይጠን ይጠን ይጠን ይጠን ይጠን ይጠን ይጠን ይጠን ይጠን ይጠን ይጠን ይጠን ይጠን ይጠን ይጠን ይጠን ይጠን ይጠን ይጠን ይጠን ይጠን ይጠን ይጠን ይጠን ይጠን ይጠን ይጠን ይጠን ይጠን ይጠን ይጠን ይጠን ይጠን ይጠን ይጠን ይጠን ይጠን ይጠን ይጠን ይጠን ይጠን ይጠን ይጠን ይጠን ይጠን ይጠን ይጠን ይጠን ይጠን ይጠን ይጠን ይጠン
The Benefits of Pixels

PIXEL can **process anything that can be rendered**
→ **Open vocabulary** which is easily extensible to **unseen text**
→ Support all written languages

**Complete parameter sharing** from the input representation
(Unlike separate-but-related subwords in an embedding matrix)

**Nothing language-specific** in the **input / output**
→ **Greater flexibility** to process written language in **different forms**
(PDFs, scanned newspapers, etc.)
Experiments
## Pretraining

<table>
<thead>
<tr>
<th>Dataset</th>
<th>Masking</th>
<th>Max. Seq. Length</th>
<th>Compute</th>
<th>Parameters</th>
</tr>
</thead>
<tbody>
<tr>
<td></td>
<td></td>
<td></td>
<td></td>
<td>86M Encoder + 26M Decoder</td>
</tr>
<tr>
<td></td>
<td></td>
<td></td>
<td></td>
<td>8x 40GB A100 GPUs for ~8 days</td>
</tr>
</tbody>
</table>

There is only **0.05% non-English** text in our **pretraining data** (estimated by Blevins and Zettlemoyer 2022)

The **Great Wall of China** *(traditional Chinese: 萬里長城; simplified Chinese: 万里长城; pinyin: Wànlǐ Chángchéng)*
Finetuning Experiments

Datasets

- [Dataset 1](https://huggingface.co/Team-PIXEL)
# Finetuning Experiments

![Hugging Face logo](https://huggingface.co/Team-PIXEL)

<table>
<thead>
<tr>
<th>Datasets</th>
<th></th>
</tr>
</thead>
<tbody>
<tr>
<td><strong>PIXEL</strong></td>
<td>English Wiki + BookCorpus</td>
</tr>
<tr>
<td><strong>BERT</strong></td>
<td>English Wiki + BookCorpus</td>
</tr>
<tr>
<td><strong>CANINE-C</strong></td>
<td>Wiki in 104 languages</td>
</tr>
</tbody>
</table>
Dependency Parsing Results

BERT UNK vs. PIXEL

- ENG: 94% BERT, 94% PIXEL
- ARA: 33% BERT, 33% PIXEL
- COP: 46% BERT, 46% PIXEL
- HIN: 85% BERT, 85% PIXEL
- JPN: 82% BERT, 82% PIXEL
- KOR: 5% BERT, 73% PIXEL
- TAM: 5% BERT, 1% PIXEL
- VIE: 5% BERT, 5% PIXEL
- ZHO: 73% BERT, 73% PIXEL

PIXEL (vastly) outperforms BERT on unseen scripts
GLUE Results

BERT outperforms PIXEL on English sentence-level tasks
Robustness against orthographic attacks (Zeroé)

<table>
<thead>
<tr>
<th>Attack</th>
<th>Sentence</th>
</tr>
</thead>
<tbody>
<tr>
<td>NONE</td>
<td>Penguins are designed to be streamlined</td>
</tr>
</tbody>
</table>
Robustness against orthographic attacks (Zeroé)

<table>
<thead>
<tr>
<th>Attack</th>
<th>Sentence</th>
</tr>
</thead>
<tbody>
<tr>
<td>NONE</td>
<td>Penguins are designed to be streamlined</td>
</tr>
</tbody>
</table>
PIXEL is more robust than BERT
Conclusions

**PIXEL** is a new type of language model that renders **text as images** instead of splitting text into a finite set of tokens.

**Rendered text** makes it possible to achieve **high-quality transfer** to **unseen scripts** in syntactic and semantic tasks.

Pixel-based learning could be a **promising research direction** to make **NLP technology accessible** to more people.
PIXEL Resources

https://github.com/xplip/pixel

https://huggingface.co/Team-PIXEL