TEXNet Project

Alex and Blake

The im2latex problem

OpenAI request for research

Implement an attention model that takes an image of a PDF math formula, and outputs the characters of the LaTeX source that generates the formula.

Getting Started

For a quick start, download a prebuilt dataset or use these tools to build your own dataset. Alternatively, you can proceed manually with the following steps:

- Download a large number papers from arXiv. There is a collection of 29,000 arXiv papers that you could get started with. It is likely that this set of 29,000 papers may contain several hundred thousand formulas, which is more than enough for getting started. As the bandwidth of arXiv is limited, it is important to be mindful of their constraints and to not write crawlers to download all the papers of arxiv.
- Use a heuristic to find all the LaTeX formulas in the LaTeX source. It can be done by looking for the text that lies between `\begin{equation}` and `\end{equation}`. Here is a list of some of the places where equations can appear in latex files. Additional examples can be found here. It is likely that even a simple heuristic for extracting latex formulas should produce in excess of 100,000 equations; if not, keep refining the heuristic.
- Compile images of all the formulas. To keep track of the correspondence between the latex formulas and their images, it is easiest to place exactly one formula on each page. Then, when processing the latex file, it is easy to keep track of the pages. Be sure to not render formulas so large that they exceed an entire page. Also, be sure to render the formulas in several fonts.
- Train a visual attention sequence-to-sequence model (as in the Show, Attend, and Tell paper, or perhaps a different variant of visual attention) that would take an image of a formula as input, and output the latex source of the formula, one character at a time. A Theano implementation of the Show, Attend, and Tell paper can help you get started. If you wish to implement your model from scratch, TensorFlow can be a good starting point.
- It takes some effort to correctly implement a sequence-to-sequence model with attention. To debug your model, we recommend that you start with a toy synthetic OCR problem, where the inputs are long images that are obtained by concatenating sequences of images of MNIST digits, and the labels should be a sequence of their classifications. While this problem can be solved without an attention model, it is useful as a sanity check, to ensure that the implementation is not badly broken.
- We recommend trying the Adam optimizer.

Proposed Model

- Neural Network

Proposed Model

- Neural Network
- Recurrent Neural Network (RNN)

Proposed Model

- Neural Network
- Recurrent Neural Network (RNN)
- Long-Short-Term-Memory (LSTM)

Proposed Model

- Neural Network
- Recurrent Neural Network (RNN)
- Long-Short-Term-Memory (LSTM)
- attention mechanism

im2latex Deng et al. 2017

- introduced im2latex-100k dataset

`im2latex` Deng et al. 2017

- introduced `im2latex-100k` dataset
- matched formulae using regular expressions, only

`im2latex` Deng et al. 2017

- introduced `im2latex-100k` dataset
- matched formulae using regular expressions, only
- normalized formulae

`im2latex` Deng et al. 2017

- introduced `im2latex-100k` dataset
- matched formulae using regular expressions, only
- normalized formulae
- BiLingual Evaluation Understudy (BLEU) of 87.73

Other Solutions to `im2latex`

Researchers	BLEU Score (%)	Training Time
Deng et al 2017	87.73	20 hours
Genthial 2017	88.00	-
Wang, Sun & Wang 2018	88.25	-
Singh 2018	89.00	60 hours
Wang & Liu 2019	90.28	75 hours

Singh's `im2latex`

Table 2: Test results. Im2latex-100k results are from Deng et al. [6]. The last column is the percentage of successfully rendering predictions.

Dataset	Model	BLEU Score	Edit Distance	Visual Match[103]	Compiling Predictions
I2L-140K	I2L-NOPOOL	**89.0%**	0.0676	70.37%	99.94%
	I2L-STRIPS	**89.0%**	0.0671	69.24%	99.85%
Im2latex-90k	I2L-STRIPS	**88.19%**	0.0725	68.03%	99.81%
Im2latex-100k	IM2TEX	87.73%	-	**79.88%**	-

Figure 2: Performance across datasets

Comments

3.2 Dataset

Datasets were created from single-line LaTeX math formulas extracted from scientific papers and subsequently processed as follows: 1) Normalize the formulas to minimize spurious ambiguity.[12] 2) Render the normalized formulas using pdflatex and discard ones that didn't compile or render successfully. 3) Remove duplicates. 4) Remove formulas with low-frequencey words (frequency-threshold = 24 for Im2latex-90k and 50 for I2l-140K). 5) Remove images bigger than 1086×126 and formulas longer than 150. Processing the Im2latex-100k dataset[104] (103559 samples) as above resulted in the **Im2latex-90k** dataset which has 93741 samples. Of these, 4648 were set aside as the test dataset and the remaining 89093 were split into training (95%) and validation (5%) sets before each run (section 2.3). We found the Im2latex-90k dataset too small for good generalization and therefore augmented it with additional samples from KDD Cup 2003. This resulted in the **I2L-140K** dataset with 114406 (training), 14280 (validation) and 14280 (test) samples. Since the normalized formulas are already space separated token sequences, no additional tokenization step was necessary. The vocabulary was therefore produced by simply identifying the set of unique space-separated words in the dataset.

Figure 3: Comments by Singh

Wang & Liu's `im2latex`

C. Impact of the Formula Length on Performance

In Figure 4, we analyze the impact of formula length on the image absolute accuracy of our neural architecture. As expected, we experience a drop in performance for longer MEs, *e.g.* for formulas under 100 tokens we achieve an accuracy of 80%, while for MEs over 150 tokens we obtain around 20%. We also note that very short formulas (*i.e.* shorter than 15 tokens) have a worse performance than medium-sized one (*i.e.* between 15 and 30 characters). The reason for these results is probably due to the imbalance in the training data, since short formulas are by far less frequent (under 3% of MEs) than the longer ones (over 13% of MEs have a length between 15 and 30 tokens).

Figure 4: Performance by number of tokens Wang & Liu

Our Data

We have

- Harvested 1 month of publications Jan 2018[1]

We want

[1] http://archive.org/

Our Data

We have

- Harvested 1 month of publications Jan 2018[1]
- Extracted All TeX files

We want

[1] http://archive.org/

Our Data

We have

- Harvested 1 month of publications Jan 2018[1]
- Extracted All TeX files
- Expanded macros in 7200 files

We want

[1] http://archive.org/

Our Data

We have

- Harvested 1 month of publications Jan 2018[1]
- Extracted All TeX files
- Expanded macros in 7200 files
- Extracted 15,220 LaTeX files

We want

[1] http://archive.org/

Our Data

We have

- Harvested 1 month of publications Jan 2018[1]
- Extracted All TeX files
- Expanded macros in 7200 files
- Extracted 15,220 LaTeX files

We want

- higher matching rate

[1] http://archive.org/

Our Data

We have

- Harvested 1 month of publications Jan 2018[1]
- Extracted All TeX files
- Expanded macros in 7200 files
- Extracted 15,220 LaTeX files

We want

- higher matching rate
- class balance

[1] http://archive.org/

Our Data

We have

- Harvested 1 month of publications Jan 2018[1]
- Extracted All TeX files
- Expanded macros in 7200 files
- Extracted 15,220 LaTeX files

We want

- higher matching rate
- class balance
- token length balance

[1] http://archive.org/

Singh's `im2latex` Model

Upgrading

- Ported model to Python 3[2]

[2] https://github.com/untrix/im2latex

Singh's `im2latex` Model

Upgrading

- Ported model to Python 3[2]
- Future plans to upgrade Tensorflow to v2

[2] https://github.com/untrix/im2latex

Singh's `im2latex` Model

Upgrading

- Ported model to Python 3[2]
- Future plans to upgrade Tensorflow to v2
- Deal with deprecated dependencies

[2] https://github.com/untrix/im2latex

Progress so far

Training model

- Model has been deployed on a Google-Cloud GPU learner and locally for consistency

Following Upgrades to Model

Progress so far

Training model

- Model has been deployed on a Google-Cloud GPU learner and locally for consistency
- Trains without problems - all log files show promising results

Following Upgrades to Model

Progress so far

Training model

- Model has been deployed on a Google-Cloud GPU learner and locally for consistency
- Trains without problems - all log files show promising results
- Currently has problems restoring from checkpoint

Following Upgrades to Model

Progress so far

Training model

- Model has been deployed on a Google-Cloud GPU learner and locally for consistency
- Trains without problems - all log files show promising results
- Currently has problems restoring from checkpoint

Following Upgrades to Model

- Be able to tweak the attention model and customizable hyperparameters of the model

Progress so far

Training model

- Model has been deployed on a Google-Cloud GPU learner and locally for consistency
- Trains without problems - all log files show promising results
- Currently has problems restoring from checkpoint

Following Upgrades to Model

- Be able to tweak the attention model and customizable hyperparameters of the model
- Upgrade preproccessing scripts to work on Blake's generated data for consistency

Progress so far

Training model

- Model has been deployed on a Google-Cloud GPU learner and locally for consistency
- Trains without problems - all log files show promising results
- Currently has problems restoring from checkpoint

Following Upgrades to Model

- Be able to tweak the attention model and customizable hyperparameters of the model
- Upgrade preproccessing scripts to work on Blake's generated data for consistency
- Experiment with running the model on various data categories (i.e Matrices, Equations, Piece-wise Functions)

Timeline

- Finalize Porting (3 days)

Timeline

- Finalize Porting (3 days)
- Initial Assessments (4 days)

Timeline

- Finalize Porting (3 days)
- Initial Assessments (4 days)
- Final Runs