Final Report for EE447

Xu Jiayu 517030910367 and Hua Zeyu 517030910277

***Abstract*—Nowadays, we university students have a large demand for reading academic papers and 95 percent of these are stored in PDF files. Our group want to simplify the procedure of reading papers, so we design a tool that helps parsing the PDF file and extracting the information. In detail, we achieve such functions: We first parse the PDF file and get the page contents of it. Next we can extract the abstract, reference list from a paper, and then automatically download these references. Corely, we extract corresponding keywords cited by the original paper so that we can locate them quickly in the referenceto broaden the wanting knowledge instead of reading the whole reference blindly. Finally, we integrate all these functions and visualize them on a web page. All the codes have been uploaded to: `https://github.com/bilkerrr/EE447_FinalProject`.**

Attention: in this report, the writer of each section also implements its function.

I. Introduction (written by 517030910367 Xu Jiayu)

We choose **PDF Parsing and Information Extraction** topic as our project. The motivation of this selection is that we are looking forward to improving the experience in reading academic papers stored in PDF file. We are always reading academic papers to broaden our minds and to keep pace with the technique development. However, tens-of-pages long and filled with quantities of reference, paper contents may make our lost in the paper. Therefore, we want to take avantage of PDF parsing tool to extract the key information of a paper so as to simplify the reading process and save time. In this project, we accomplish the following functions: (1) parse PDF file, (2) extract abstract of a paper, (3) extract reference list from a paper, (4) automatically downaload these references, (5) extract the key concepts of a reference cited by the original paper, (6) highlight these keywords in the reference PDF, (7) integrate all above functions and visualize them on a web page.

II. Preliminaries (written by 517030910367 Xu Jiayu)

A. PDF Background

Portable Document Format (PDF) is a file format developed by Adobe in the 1990s to present documents including texts and images. Based on the `PostScript` language, each PDF file encapsulates a complete description of a fixed-layout flat document whose structure is shown in 1, including the texts, fonts, vector graphics and other information. Howver PDF has no logical structures like sentences or paragraphs. Therefore, in order to get the real contents of a PDF we need to do further processing, called PDF parsing.

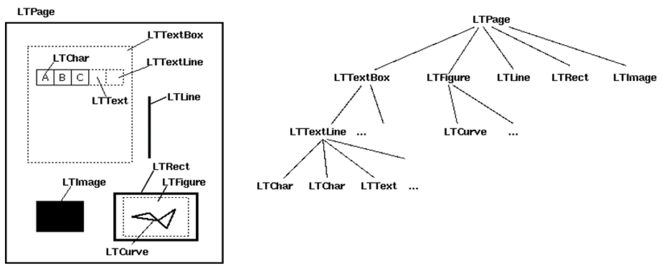


Fig. 1. Tree Structure of a PDF File

B. PDFMiner Tool

PDFMiner[1] is a text extraction tool for PDF documents. It takes a strategy of lazy parsing, only to parse the stuff when needed, and hence it can save time and memory. To implement the parsing, it takes advantage of several classes shown in 2.

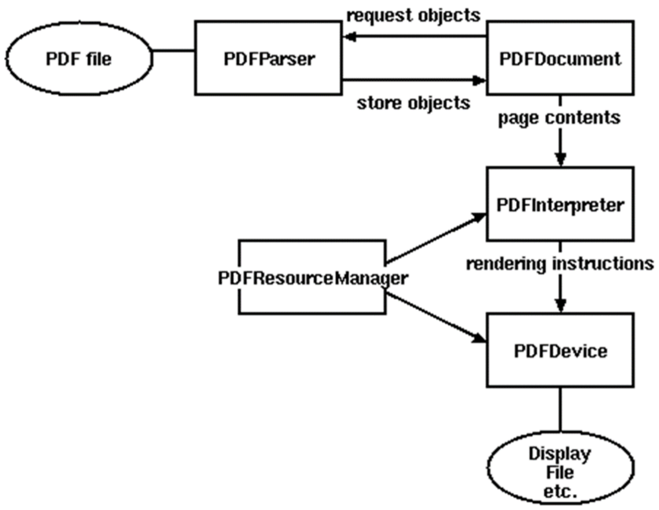


Fig. 2. Relationships between PDFMiner Classes

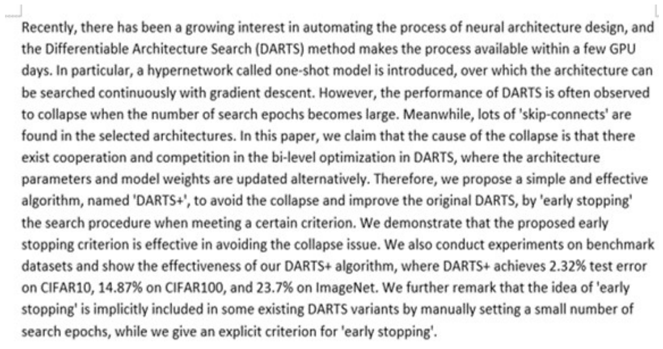
`PDFParse` fetches data from a file and `PDFDocument` stores it. Then `PDFPageInterpreter` to process the page co-tents and `PDFDevice` translate it into the layout structure shown in 1. In our project, we can call the `LTTextBox` nodes to get the text information.

C. *RAKE Algorithm*

Rapid Automatic Keyword Extraction (RAKE)[2] is proposed by Alyona Medelyan. The basic idea of the algorithm is that first to make phrase segmentation, score each word in the phrase, and rank the phrases with the sum score. To score each word, it first establishes co-present matrix for each word and calculates the degree in the matrix divided by the word frequency as its score. It works well in extracting jargons and runs fast.

III. PDF Parsing (written by 517030910367 Xu Jiayu)

In this part we use PDFMiner to get the text content of a PDF file. But though PDFMiner provides us with a convenient API, we still need further check and revise. For example, there might exist a link break in a word and a '-' will be attached to the end of a line such that, to restore the word, we need to remove the string `'-\n'`. What's more, some encoding problems need to be paid attention to like 'fi'.
The result is shown in 3.



Recently, there has been a growing interest in automating the process of neural architecture design, and the Differentiable Architecture Search (DARTS) method makes the process available within a few GPU days. In particular, a hypernetwork called one-shot model is introduced, over which the architecture can be searched continuously with gradient descent. However, the performance of DARTS is often observed to collapse when the number of search epochs becomes large. Meanwhile, lots of 'skip-connects' are found in the selected architectures. In this paper, we claim that the cause of the collapse is that there exist cooperation and competition in the bi-level optimization in DARTS, where the architecture parameters and model weights are updated alternatively. Therefore, we propose a simple and effective algorithm, named 'DARTS+', to avoid the collapse and improve the original DARTS, by 'early stopping' the search procedure when meeting a certain criterion. We demonstrate that the proposed early stopping criterion is effective in avoiding the collapse issue. We also conduct experiments on benchmark datasets and show the effectiveness of our DARTS+ algorithm, where DARTS+ achieves 2.32% test error on CIFAR10, 14.87% on CIFAR100, and 23.7% on ImageNet. We further remark that the idea of 'early stopping' is implicitly included in some existing DARTS variants by manually setting a small number of search epochs, while we give an explicit criterion for 'early stopping'.

Fig. 3. The Result of Abstract Extraction

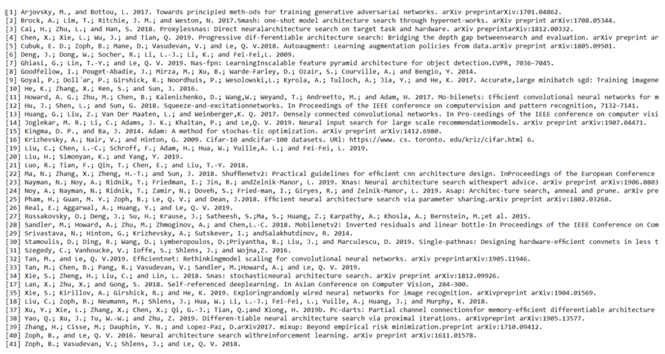
IV. Information Extraction (written by 517030910367 Xu Jiayu)

A. *Abstract Extraction*

It is quite easy to implement. We only need to search for the independent paragraph `Abstract` and the next paragraph is what we are looking for.

B. *Reference List Extraction*

Since reference are ordered in a special format, e.g. "[number] author name. paper title ...", we can take advantage of regular expression to fetch them. The result is shown in 4.



```
[1] Arjovsky, M., and Bottou, L. 2017. Towards principled meth-ods for training generative adversarial networks. arXiv preprintarXiv:1701.04862.
[2] Brock, A.; Lim, T.; Ritchie, J. M.; and Weston, N. 2017.Smash: one-shot model architecture search through hypernet-works. arXiv preprint arXiv:1708.05344.
[3] Cai, H.; Zhu, L.; and Han, S. 2018. Proxylessnas: Direct neuralarchitecture search on target task and hardware. arXiv preprintarXiv:1812.00332.
[4] Chen, X.; Xie, L.; Wu, J.; and Tian, Q. 2019. Progressive dif-ferentiable architecture search: Bridging the depth gap betweensearch and evaluation. arXiv preprint ar
[5] Cubuk, E. D.; Zoph, B.; Mane, D.; Vasudevan, V.; and Le, Q. V.2018. Autoaugment: Learning augmentation policies from data.arXiv preprint arXiv:1805.09501.
[6] Deng, J.; Dong, W.; Socher, R.; Li, L.-J.; Li, K.; and Fei-Fei,L. 2009.
[7] Ghiasi, G.; Lin, T.-Y.; and Le, Q. V. 2019. Nas-fpn: LearningInscalable feature pyramid architecture for object detection.CVPR, 7036–7045.
[8] Goodfellow, I.; Pouget-Abadie, J.; Mirza, M.; Xu, B.; Warde-Farley, D.; Ozair, S.; Courville, A.; and Bengio, Y. 2014.
[9] Goyal, P.; Doll´ar, P.; Girshick, R.; Noordhuis, P.; Wesolowski,L.; Kyrola, A.; Tulloch, A.; Jia, Y.; and He, K. 2017. Accurate,large minibatch sgd: Training imagene
[10] He, K.; Zhang, X.; Ren, S.; and Sun, J. 2016.
[11] Howard, A. G.; Zhu, M.; Chen, B.; Kalenichenko, D.; Wang,W.; Weyand, T.; Andreetto, M.; and Adam, H. 2017. Mo-bilenets: Efficient convolutional neural networks for m
[12] Hu, J.; Shen, L.; and Sun, G. 2018. Squeeze-and-excitationnetworks. In Proceedings of the IEEE conference on computervision and pattern recognition, 7132–7141.
[13] Huang, G.; Liu, Z.; Van Der Maaten, L.; and Weinberger,K. Q. 2017. Densely connected convolutional networks. In Pro-ceedings of the IEEE conference on computer visi
[14] Joglekar, M. R.; Li, C.; Adams, J. K.; Khaitan, P.; and Le,Q. V. 2019. Neural input search for large scale recommendationmodels. arXiv preprint arXiv:1907.04471.
[15] Kingma, D. P., and Ba, J. 2014. Adam: A method for stochas-tic optimization. arXiv preprint arXiv:1412.6980.
[16] Krizhevsky, A.; Nair, V.; and Hinton, G. 2009. Cifar-10 andcifar-100 datasets. URl: https://www. cs. toronto. edu/kriz/cifar.html 6.
[19] Liu, C.; Chen, L.-C.; Schroff, F.; Adam, H.; Hua, W.; Yuille,A. L.; and Fei-Fei, L. 2019.
[20] Liu, H.; Simonyan, K.; and Yang, Y. 2019.
[21] Luo, R.; Tian, F.; Qin, T.; Chen, E.; and Liu, T.-Y. 2018.
[22] Ma, N.; Zhang, X.; Zheng, H.-T.; and Sun, J. 2018. Shufflenetv2: Practical guidelines for efficient cnn architecture design. InProceedings of the European Conference
[23] Nayman, N.; Noy, A.; Ridnik, T.; Friedman, I.; Jin, R.; andZelnik-Manor, L. 2019. Xnas: Neural architecture search withexpert advice. arXiv preprint arXiv:1906.0803
[24] Noy, A.; Nayman, N.; Ridnik, T.; Zamir, N.; Doveh, S.; Fried-man, I.; Giryes, R.; and Zelnik-Manor, L. 2019. Asap: Architec-ture search, anneal and prune. arXiv pre
[25] Pham, H.; Guan, M. Y.; Zoph, B.; Le, Q. V.; and Dean, J.2018. Efficient neural architecture search via parameter sharing.arXiv preprint arXiv:1802.03268.
[26] Real, E.; Aggarwal, A.; Huang, Y.; and Le, Q. V. 2019.
[27] Russakovsky, O.; Deng, J.; Su, H.; Krause, J.; Satheesh, S.;Ma, S.; Huang, Z.; Karpathy, A.; Khosla, A.; Bernstein, M.;et al. 2015.
[28] Sandler, M.; Howard, A.; Zhu, M.; Zhmoginov, A.; and Chen,L.-C. 2018. Mobilenetv2: Inverted residuals and linear bottle-In Proceedings of the IEEE Conference on Com
[29] Srivastava, N.; Hinton, G.; Krizhevsky, A.; Sutskever, I.; andSalakhutdinov, R. 2014.
[30] Stamoulis, D.; Ding, R.; Wang, D.; Lymberopoulos, D.;Priyantha, B.; Liu, J.; and Marculescu, D. 2019. Single-pathnas: Designing hardware-efficient convnets in less t
[31] Szegedy, C.; Vanhoucke, V.; Ioffe, S.; Shlens, J.; and Wojna,Z. 2016.
[32] Tan, M., and Le, Q. V.2019. Efficientnet: Rethinkingmodel scaling for convolutional neural networks. arXiv preprintarXiv:1905.11946.
[33] Tan, M.; Chen, B.; Pang, R.; Vasudevan, V.; Sandler, M.;Howard, A.; and Le, Q. V. 2019.
[34] Xie, S.; Zheng, H.; Liu, C.; and Lin, L. 2018. Snas: stochasticneural architecture search. arXiv preprint arXiv:1812.09926.
[17] Lan, X.; Zhu, X.; and Gong, S. 2018. Self-referenced deeplearning. In Asian Conference on Computer Vision, 284–300.
[35] Xie, S.; Kirillov, A.; Girshick, R.; and He, K. 2019. Exploringrandomly wired neural networks for image recognition. arXivpreprint arXiv:1904.01569.
[18] Liu, C.; Zoph, B.; Neumann, M.; Shlens, J.; Hua, W.; Li, L.-J.; Fei-Fei, L.; Yuille, A.; Huang, J.; and Murphy, K. 2018.
[37] Xu, Y.; Xie, L.; Zhang, X.; Chen, X.; Qi, G.-J.; Tian, Q.;and Xiong, H. 2019b. Pc-darts: Partial channel connectionsfor memory-efficient differentiable architecture
[38] Yao, Q.; Xu, J.; Tu, W.-W.; and Zhu, Z. 2019. Differen-tiable neural architecture search via proximal iterations. arXivpreprint arXiv:1905.13577.
[39] Zhang, H.; Cisse, M.; Dauphin, Y. N.; and Lopez-Paz, D.arXiv2017. mixup: Beyond empirical risk minimization.preprint arXiv:1710.09412.
[40] Zoph, B., and Le, Q. V. 2016. Neural architecture search withreinforcement learning. arXiv preprint arXiv:1611.01578.
[41] Zoph, B.; Vasudevan, V.; Shlens, J.; and Le, Q. V. 2018.
```

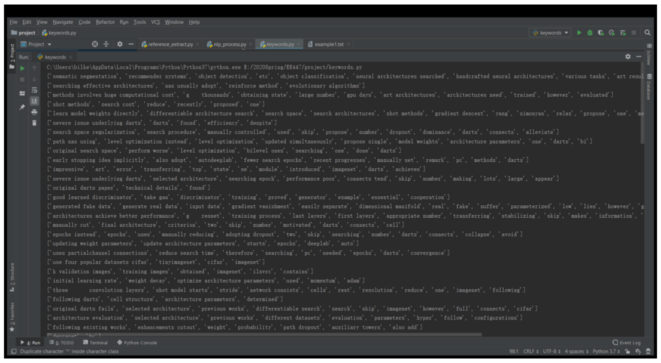
Fig. 4. The Result of Reference List Extraction

C. *Reference Downloading*

Considering the patent problem, we only provide with downloading in ArXiv website. The implementation also makes use of regular expression to fetch the ArXiv ID of paper that recorded in the reference information. Then we can automatically generate the PDF URL according to the ArXiv ID for downloading.

D. Keyword Extraction

We first need to do sentence tokenize and only preserve the sentences with citations. Sentence tokenize can be implemented according to the punctuations and Captial letters in the beginning of the sentence. Then we use RAKE algorithm to extract keywords of each sentence. After that, we match the keywords with corresponding reference. We split sentence into short clauses by puncuation marks, and then select the **nearest neighbour** citation number as the result of keyword matching. The intermediate keyword extraction result is shown in 5.

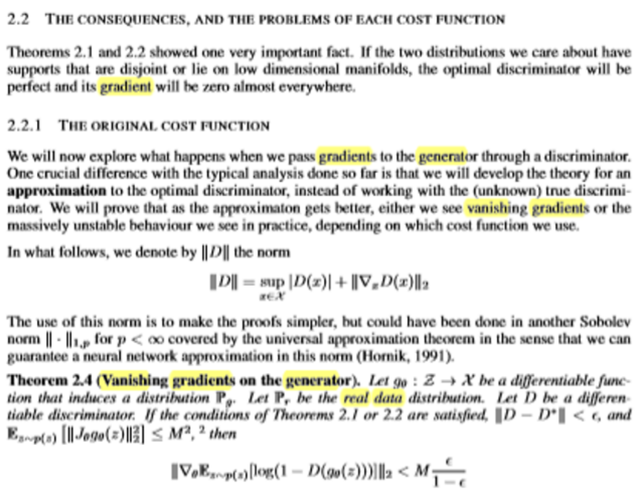

File Edit View Navigate Code Refactor Run Tools VCS Window Help
project
keywords.py
Project
reference_extract.py
nlp_process.py
keywords.py
example1.txt
keywords
Run:
4: Run
6: TODO
Terminal
Python Console
Event Log
Duplicate character '^' inside character class
98:1 CRLF UTF-8 4 spaces Python 3.7

```
C:\Users\bilke\AppData\Local\Programs\Python\Python37\python.exe E:/2020Spring/EE447/project/keywords.py
['semantic segmentation', 'recommender systems', 'object detection', 'etc', 'object classification', 'neural architectures searched', 'handcrafted neural architectures', 'various tasks', 'art resul
['searching effective architectures', 'nas usually adopt', 'reinforce method', 'evolutionary algorithms']
['methods involves huge computational cost', 'g    thousands', 'obtaining state', 'large number', 'gpu days', 'art architectures', 'architectures need', 'trained', 'however', 'evaluated']
['shot methods', 'search cost', 'reduce', 'recently', 'proposed', 'one']
['learn model weights directly', 'differentiable architecture search', 'search space', 'search architectures', 'shot methods', 'gradient descent', 'yang', 'simonyan', 'relax', 'propose', 'one', 'me
['severe issue underlying darts', 'darts', 'found', 'efficiency', 'despite']
['search space regularization', 'search procedure', 'manually controlled', 'used', 'skip', 'propose', 'number', 'dropout', 'dominance', 'darts', 'connects', 'alleviate']
['path nas using', 'level optimization instead', 'level optimization', 'updated simultaneously', 'propose single', 'model weights', 'architecture parameters', 'one', 'darts', 'bi']
['original search space', 'perform worse', 'level optimization', 'bilevel ones', 'searching', 'one', 'done', 'darts']
['early stopping idea implicitly', 'also adopt', 'autodeeplab', 'fewer search epochs', 'recent progresses', 'manually set', 'remark', 'pc', 'methods', 'darts']
['impressive', 'art', 'error', 'transferring', 'top', 'state', 'se', 'module', 'introduced', 'imagenet', 'darts', 'achieves']
['severe issue underlying darts', 'selected architecture', 'searching epoch', 'performance poor', 'connects tend', 'skip', 'number', 'making', 'lots', 'large', 'appear']
['original darts paper', 'technical details', 'found']
['good learned discriminator', 'take gan', 'discriminator', 'training', 'proved', 'generator', 'example', 'essential', 'cooperation']
['generated fake data', 'generate real data', 'input data', 'gradient vanishment', 'easily separate', 'dimensional manifold', 'real', 'fake', 'suffer', 'parameterized', 'low', 'lies', 'however', 'g
['architectures achieve better performance', 'g    resnet', 'training process', 'last layers', 'first layers', 'appropriate number', 'transferring', 'stabilizing', 'skip', 'makes', 'information', '
['manually cut', 'final architecture', 'criterion', 'two', 'skip', 'number', 'motivated', 'darts', 'connects', 'cell']
['epochs instead', 'epochs', 'uses', 'manually reducing', 'adopting dropout', 'two', 'skip', 'searching', 'number', 'darts', 'connects', 'collapse', 'avoid']
['updating weight parameters', 'update architecture parameters', 'starts', 'epochs', 'deeplab', 'auto']
['uses partialchannel connections', 'reduce search time', 'therefore', 'searching', 'pc', 'needed', 'epochs', 'darts', 'convergence']
['use four popular datasets cifar', 'tinyimagenet', 'cifar', 'imagenet']
['k validation images', 'training images', 'obtained', 'imagenet', 'ilsvrc', 'contains']
['initial learning rate', 'weight decay', 'optimize architecture parameters', 'used', 'momentum', 'adam']
['three      convolution layers', 'shot model starts', 'stride', 'network consists', 'cells', 'rest', 'resolution', 'reduce', 'one', 'imagenet', 'following']
['following darts', 'cell structure', 'architecture parameters', 'determined']
['original darts fails', 'selected architecture', 'previous works', 'differentiable search', 'search', 'skip', 'imagenet', 'however', 'full', 'connects', 'cifar']
['architecture evaluation', 'selected architecture', 'previous works', 'different datasets', 'evaluation', 'parameters', 'hyper', 'follow', 'configurations']
['following existing works', 'enhancements cutout', 'weight', 'probability', 'path dropout', 'auxiliary towers', 'also add']
```

Fig. 5. The Result of RAKE Algorithm

E. Reference Highlighting

To search the keyword in the reference paper, we first need to process the keywords, because the phrase expression in each paper may vary. For example, the original paper says "gradient vanishment" but the reference uses the expression of "vanishing gradient". We need to extract stem of each keyword. Three common stemmers are widely used: PorterStemmer, LancasterStemmer and SnowballStemmer. After test, we choose SnowballStemmer due to its outperformance. Now, we want to highlight the keywords in the PDF. We use another PDF parsing tool: MuPDF[3]. It stands out among all similar products for its top rendering, and it can do more than PDFMiner. Here we use two functions: `searchFor` to locate the keywords and `addHighlightAnnot` to highlight them.
A sample result is shown in 6. The original keywords are: **generator**, **real data** and **gradient vanishment**.



2.2 THE CONSEQUENCES, AND THE PROBLEMS OF EACH COST FUNCTION

Theorems 2.1 and 2.2 showed one very important fact. If the two distributions we care about have supports that are disjoint or lie on low dimensional manifolds, the optimal discriminator will be perfect and its gradient will be zero almost everywhere.

2.2.1 THE ORIGINAL COST FUNCTION

We will now explore what happens when we pass gradients to the generator through a discriminator. One crucial difference with the typical analysis done so far is that we will develop the theory for an **approximation** to the optimal discriminator, instead of working with the (unknown) true discriminator. We will prove that as the approximaton gets better, either we see vanishing gradients or the massively unstable behaviour we see in practice, depending on which cost function we use.

In what follows, we denote by $\|D\|$ the norm

$$\|D\| = \sup_{x\in\mathcal{X}} |D(x)| + \|\nabla_x D(x)\|_2$$

The use of this norm is to make the proofs simpler, but could have been done in another Sobolev norm $\|\cdot\|_{1,p}$ for $p < \infty$ covered by the universal approximation theorem in the sense that we can guarantee a neural network approximation in this norm (Hornik, 1991).

Theorem 2.4 (Vanishing gradients on the generator). *Let $g_\theta : \mathcal{Z} \to \mathcal{X}$ be a differentiable function that induces a distribution $\mathbb{P}_g$. Let $\mathbb{P}_r$ be the real data distribution. Let D be a differentiable discriminator. If the conditions of Theorems 2.1 or 2.2 are satisfied, $\|D - D^*\| < \epsilon$, and $\mathbb{E}_{z\sim p(z)}\left[\|J_\theta g_\theta(z)\|_2^2\right] \leq M^2$, [2] then*

$$\|\nabla_\theta \mathbb{E}_{z\sim p(z)}[\log(1 - D(g_\theta(z)))]\|_2 < M\frac{\epsilon}{1-\epsilon}$$

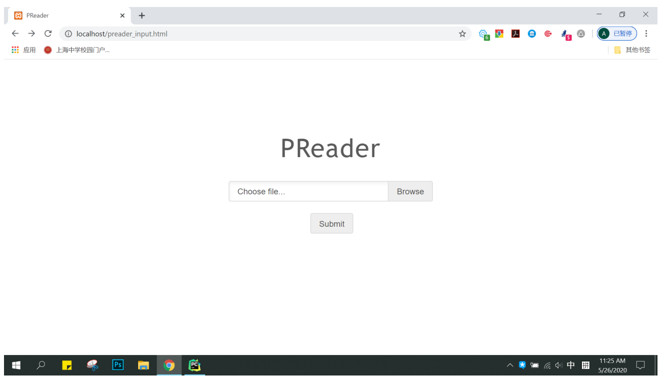
Fig. 6. Highlighter Result

V. Visualization (written by 517030910277 Hua Zeyu)

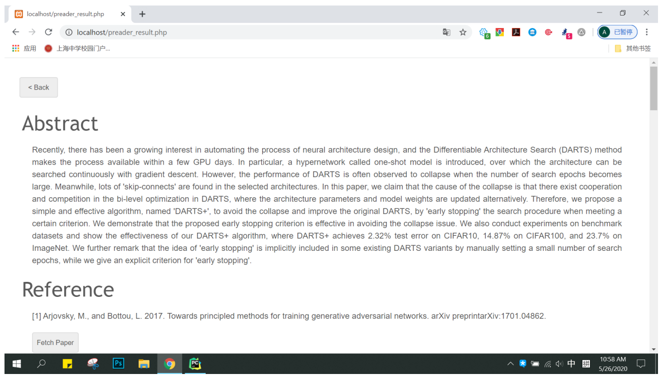
To visualize our work, we made a web application that is capable of extracting abstract and references as well as downloading highlighted referenced papers. User guide and how the application is implemented will be introduced in the following parts.

A. User Guide

The homepage looks like a search engine, where you can upload a paper and submit to the server.



After the paper is uploaded, the extracted information is showed on the page. There is a 'BACK' button on the top of the page, redirecting to the home page for the next submission. The abstract is ranged under the button.



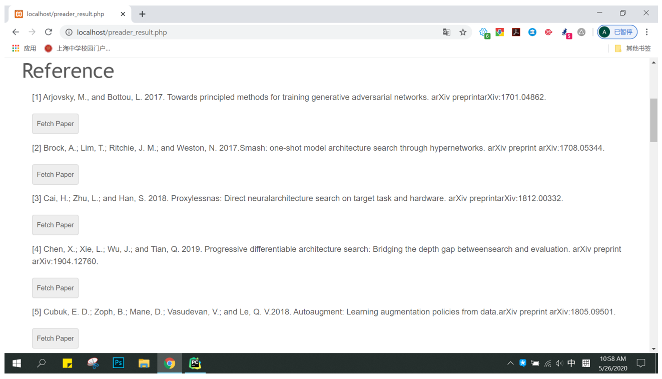
Abstract

Recently, there has been a growing interest in automating the process of neural architecture design, and the Differentiable Architecture Search (DARTS) method makes the process available within a few GPU days. In particular, a hypernetwork called one-shot model is introduced, over which the architecture can be searched continuously with gradient descent. However, the performance of DARTS is often observed to collapse when the number of search epochs becomes large. Meanwhile, lots of 'skip-connects' are found in the selected architectures. In this paper, we claim that the cause of the collapse is that there exist cooperation and competition in the bi-level optimization in DARTS, where the architecture parameters and model weights are updated alternatively. Therefore, we propose a simple and effective algorithm, named 'DARTS+', to avoid the collapse and improve the original DARTS, by 'early stopping' the search procedure when meeting a certain criterion. We demonstrate that the proposed early stopping criterion is effective in avoiding the collapse issue. We also conduct experiments on benchmark datasets and show the effectiveness of our DARTS+ algorithm, where DARTS+ achieves 2.32% test error on CIFAR10, 14.87% on CIFAR100, and 23.7% on ImageNet. We further remark that the idea of 'early stopping' is implicitly included in some existing DARTS variants by manually setting a small number of search epochs, while we give an explicit criterion for 'early stopping'.

Reference

[1] Arjovsky, M., and Bottou, L. 2017. Towards principled methods for training generative adversarial networks. arXiv preprintarXiv:1701.04862.

Scroll down the page and you will see the references. If the reference is from arXiv, there will be a 'FETCH PAPER' button under it. If you click the button, the corresponding referenced paper will be downloaded. Limited by copyright, however, we do not provide other access to papers, so non-arXiv papers cannot be downloaded from our application.



Reference

[1] Arjovsky, M., and Bottou, L. 2017. Towards principled methods for training generative adversarial networks. arXiv preprintarXiv:1701.04862.

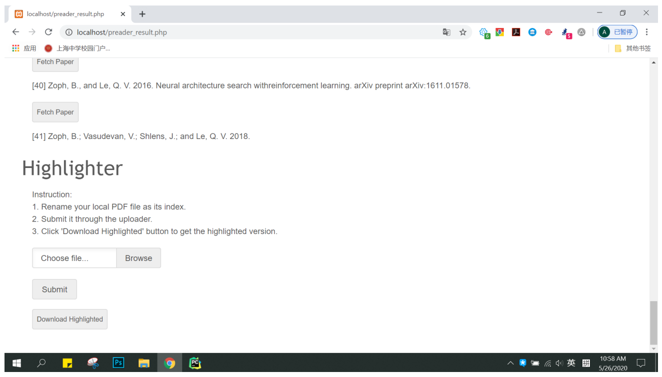
[2] Brock, A.; Lim, T.; Ritchie, J. M.; and Weston, N. 2017.Smash: one-shot model architecture search through hypernetworks. arXiv preprint arXiv:1708.05344.

[3] Cai, H.; Zhu, L.; and Han, S. 2018. Proxylessnas: Direct neuralarchitecture search on target task and hardware. arXiv preprintarXiv:1812.00332.

[4] Chen, X.; Xie, L.; Wu, J.; and Tian, Q. 2019. Progressive differentiable architecture search: Bridging the depth gap betweensearch and evaluation. arXiv preprint arXiv:1904.12760.

[5] Cubuk, E. D.; Zoph, B.; Mane, D.; Vasudevan, V.; and Le, Q. V.2018. Autoaugment: Learning augmentation policies from data.arXiv preprint arXiv:1805.09501.

The last part is the highlighter. You can upload the referenced paper (you must renamed as its index first) and get a highlighted version. The highlighted keywords are extracted from the original paper you submit on the home page and that cites the current paper.



[40] Zoph, B., and Le, Q. V. 2016. Neural architecture search withreinforcement learning. arXiv preprint arXiv:1611.01578.

[41] Zoph, B.; Vasudevan, V.; Shlens, J.; and Le, Q. V. 2018.

Highlighter

Instruction:
1. Rename your local PDF file as its index.
2. Submit it through the uploader.
3. Click 'Download Highlighted' button to get the highlighted version.

B. *Implementation*

We deploy our project on apache server which is installed by xampp. The process of PDF analyzing and information extraction is written in Python and the web application is written in PHP, CSS, HTML and JavaScript.

1) Call Python Scripts in PHP: As is stated before, PDF analyzing and visualization parts are written in different languages, so we must find a way to connect Python to PHP.

The 'exec()' function can call Python scripts in PHP. It accepts three parameter, i.e. a string $command which is the command to be executed, an array $output that saves the result and an integer $return_var indicating the return value.

```
$path="C:[redacted]\Python\python.exe ./pyfile/information_extract.py ";
$params = "C:/xampp/htdocs/source/".$fileInfoName;

exec($path.$params, $array, $res);
```

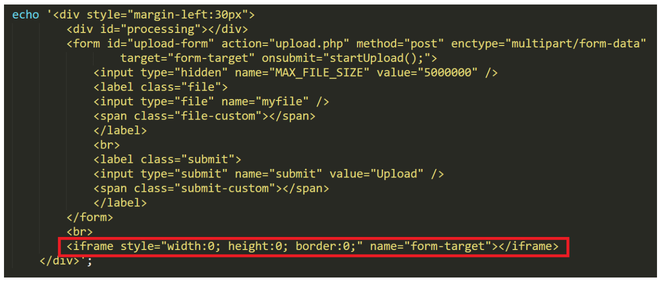
2) Upload Files: Uploading files is an important function in our application and we met most of the problems in this part.

Size constraints Since papers often contain pictures and tables, the size of PDF files is seldom less than 2MB. When we tried to upload files at first, we could successfully upload small files of hundreds of KBs, but we failed uploading files larger than 2MB. After searching some materials, we found out that this was because of the limits set in the configuration file of PHP. Here are some parmeters[5] to be revised:

- max_execution_time
- max_input_time
- post_max_size
- upload_max_filesize
- memory_limit

Upload without refreshing After we upload a paper on the home page, we are redirected to the second page. When we upload a referenced paper in the highlighter, however, we do not want to be redirected to another page, nor do we want to refresh the current page. However, since the file is uploaded through a form, there has to be a field of target.

The solution is to use iframe tag[4] in the page. iframe can embed a sub-html inside the large html. If we set target of the form as the iframe and when we try to upload a paper in highlighter, only iframe will be refreshed, but others will not. Hence, it looks like the page is not refreshed at all.



```
echo '<div style="margin-left:30px">
        <div id="processing"></div>
        <form id="upload-form" action="upload.php" method="post" enctype="multipart/form-data"
                target="form-target" onsubmit="startUpload();">
            <input type="hidden" name="MAX_FILE_SIZE" value="5000000" />
            <label class="file">
            <input type="file" name="myfile" />
            <span class="file-custom"></span>
            </label>
            <br>
            <label class="submit">
            <input type="submit" name="submit" value="Upload" />
            <span class="submit-custom"></span>
            </label>
        </form>
        <br>
        <iframe style="width:0; height:0; border:0;" name="form-target"></iframe>
    </div>';
```

VI. Future Work (written by 517030910277 Hua Zeyu)

Although we have accomplished our goals in mid-term report, there is still a long way to go.

Firstly, we are currently using existed NLP model for general use, but we can train our own model to better fit keyword extraction in academic papers.

Secondly, we use Python for PDF extraction but PHP for visualization. As suggested by teaching assistant, we can use Django to constuct our website, which is written in Python and can better cope with PDF analyzing part.

Finally, we only extract keywords in referenced papers so far, but we can try to extract key sentences later.

References

[1] https://pypi.org/project/pdfminer/

[2] Rose, Stuart & Engel, Dave & Cramer, Nick & Cowley, Wendy. (2010). Automatic Keyword Extraction from Individual Documents. 10.1002/9780470689646.ch1.

[3] https://mupdf.com/

[4] https://www.cnblogs.com/wangmeijian/p/3978407.html?tdsourcetag=s_pctim

[5] https://blog.csdn.net/sinat_34328764/article/details/80053322?tdsourcetag=s_pctim_aiomsg