Learning to Generate Lecture Posters

Yuncong Liu
Shanghai Jiao Tong University
Shanghai, China
assassain_lyc@sjtu.edu.cn

Haoyu Ling
Shanghai Jiao Tong University
Shanghai, China
smallling@sjtu.edu.cn

Luoxuan Li
Shanghai Jiao Tong University
Shanghai, China
PhoeniXLL@sjtu.edu.cn

ABSTRACT

Lecture posters are often used to promote a lecture. They can provide the information about a lecture's theme, date, and etc. However, it's difficult to generate lecture posters automatically, since we need to keep the design of them *aesthetic* and *reasonable*. In this paper, we study on how to generate a lecture poster from a background image and a text set. What we mainly need to consider is how to find an appropriate place on the background for a given text. In our method, we design two neural networks to determine the importance of a place on the background image and its harmony with text. Our generator is to arrange all the texts in the appropriate place based on some design guidelines with help of the two networks. To train our model, we create two datasets from some background images and well-designed posters collected online. The final results of the quality inspection experiments of the posters we generated show that our method is effective.

CCS Concepts

• Computing methodologies → Computer vision

Keywords

Lecture Poster Generation, Multimodal, Text Placement

1. INTRODUCTION

Posters play an important role in many aspects of our daily life, such as promoting movies, improving the popularity of enterprises and so on. Among different types of posters, the lecture poster can promote a lecture, present the description of its theme, provide the information about its date and place, etc. So automatic lecture poster generation is a very meaningful task. However, in the filed of computer vision, almost all the generation related works are about image generation and most of the methods proposed in them use GAN(Generative adversarial networks)[2]. There are only few of papers[6][7] on generating scientific posters. But in view of the particularity of scientific posters, the method proposed is difficult to be generalized to lecture posters. So we decide to do the work of learning to generate lecture posters to fill this gap.

A lecture poster consists of two parts: image and text. We simplify this problem by specifying that only one background image is required for each poster to be generated. In other words, the materials we use in generation are a background image and a text set. Therefore, we mainly need to consider about finding a suitable area for each piece of text on the background.

Our goal is to keep the posters generated *aesthetic* and ensure that they are designed *reasonably*. In order to generate a lecture poster which meets the above design principles, there are many problems we need to solve: (1) We should to be able to determine whether an area of the background is suitable for placing text. (2) For each piece of text, it should be placed in a position that reflects its importance. (3) What kind of relationship should all texts be arranged according to.

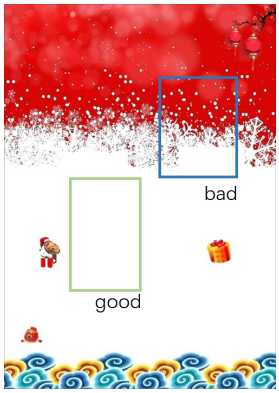
In this paper, we design a poster generator based on the methods that deal with the above three problems. For the first two problem, we design a neural network for each one and create two datasets from some background images and artificially designed posters collected online. For the last one, we add some design guidelines to the generator based on the importance of different texts. In conclusion, our lecture poster generator is to put all the texts in the appropriate place on the background image based on the above two networks and some design guidelines.

To evaluate our poster generator, we generate some lecture posters with it. Then we invite several people to score the lecture posters on *reasonability* and *aesthetics* from 0 to 10, where 0 and 10 indicate the lowest and highest scores respectively. In the end, more than 80% of people give an average score higher than 7 points. This shows that our method is effective.

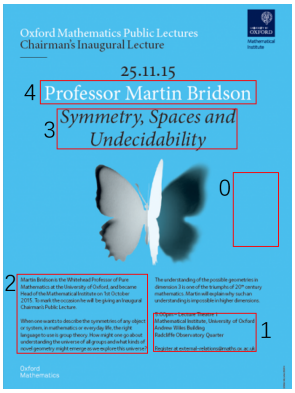
2. RELATED WORK

In this section, we review the related works in poster generation[6][7] and a related tool GAN[2]. And we show the differences between these works and out task.

Poster Generation Qiang *et al.* [6][7] try to learn to generate scientific posters from scientific papers. They extract the text content and graphical content from the paper, and then learn to generate layout for placing them. The hierarchical structure of texts depends on the paper from which it is generated. But like other works about layout



(a)Harmony



(b)Importance Level

Figure 1: Two attributes of the rectangular blocks on the background image.

generation[4][1], they are actually placing content in a black space. Because they do not need to generate a scientific paper on a background image. Therefore, they do not need to consider whether an area of the background is suitable for placing text, which is needed in our problem. And they usually need to fill the whole space in generation , while we only need to select an appropriate place for each piece of text.

GAN GAN is one of the most popular generative network in recent years[2][9]. It has generator network and discriminator network. The generator network is trained to deceive the discriminator network. In the training progress, it gradually produces more and more realistic images which means that artificial images look indistinguishable from real images and the discriminator network are getting harder to discriminate two images. At the same time, the discriminator continues to adapt to the gradual improvement of the generator, setting a high degree of realism for the generated image. Once training is complete, the generator is able to transform any point in its input space into a trusted image. However GAN is a dynamic system in which the optimization process seeks not the minimum, but the balance between the two networks. For this reason, GAN is well known to be difficult to train and making GAN work requires a lot of careful adjustment of model architecture and training parameters. In our work of generation, the main consideration is to place text on the image.

3. METHOD

Problem Setting In order to generate a *reasonable* and *aesthetic* lecture poster, we refer to the rule on how people design lecture posters in practice. A lecture poster contains text and images. We simplify this problem by specifying that only one back-ground image is required for each poster to be generated. So, for the problem, the input are a background image G and a set T of text (e.g., title, date, etc.) and the output is a lecture poster. Further, we hope that each text $t \in T$ can be placed independently. In other words, a continuous piece of text will not be split into two parts in the input set T. In this way, the main thing we need to do is to find a suitable rectangle on the background image G for each text $t \in T$.

Overview In the following subsections, we will first introduce two networks (Sec. 3.1 and Sec. 3.2) which are used to determine whether a rectangular block of the image G is suitable for a given text t. Then there is our generator (Sec. 3.3) which generates lecture posters based on these two networks and some design guidelines.

3.1 Harmony between Image and Text

Generally speaking, in an area of an image, if its color changes in a larger range, the more ups and downs, then its is less suitable for placing text. On the contrary, it is more suitable. Because in the front case, if the text is placed in this area, it will look messy, which is unfriendly to readers. We think that such areas of the image are less harmonious with the text. If the harmony between an area of image and text is higher than a certain threshold, then the area is good, i.e., it's suitable for text. Figure 1(a) shows a good rectangular area and a bad one respectively. Intuitively, we can employ a binary classifier to solve this problem.

Data Processing We adopt a tuple (x, y, w, h) to represent a rectangular block on a image, where (x, y) is the position of the upper left corner of the rectangle, and w and h are its width and height. In our method, for a given rectangle of a image, we need to extract its RGB data d of size $3 \times w \times h$ first. In order to show the degree of "messy" in this rectangle, we let

$$d_{i,j,k} = \frac{|d_{i,j,k} - D_i|}{255} \tag{1}$$

where $d_{i,j,k}$ is the value of position (j, k) of channel i in d and D_i is the average of all values of channel i in d.

Network After the above processing, d can be the input of our network. The first layer of our classifier network is a two dimensional adaptive average pooling layer. For different input sizes, it can fix the output size by adjusting its parameters. Since the rectangles selected may have different sizes, we can convert d to a fixed size $3 \times k \times k$ through it. In each dimension of this pooling layer, the parameters are:

$$\begin{aligned} l &= \lfloor \frac{s_i}{s_o} \rfloor \\ s_k &= s_i - (s_o - 1) \times l \end{aligned} \tag{2}$$

where l is stride, and s_i, s_o and s_k are input size, output size and kernel size respectively. The success of this layer are two fully connected layers connected in sequence. Then we get the final output: the probability of each category. And because this is a binary classifier and sigmoid is used as the activation function, the output of the network can be directly regarded as the harmony between the input area and text.

Since this is a binary classification problem, we use binary cross-entropy as the loss function.

3.2 Importance of Position

Different texts have different importance on a lecture poster. For example, the title is more important than additional information about the presenter. Based on human experience, we set the importance level of different texts as Table 1 where supplementary information includes the presenter's email address, live URL, etc.

We specify that each text $t \in T$ should have an input attribute p, indicating which one of the types in Table 1 it is. Intuitively, text of a higher importance should be placed in

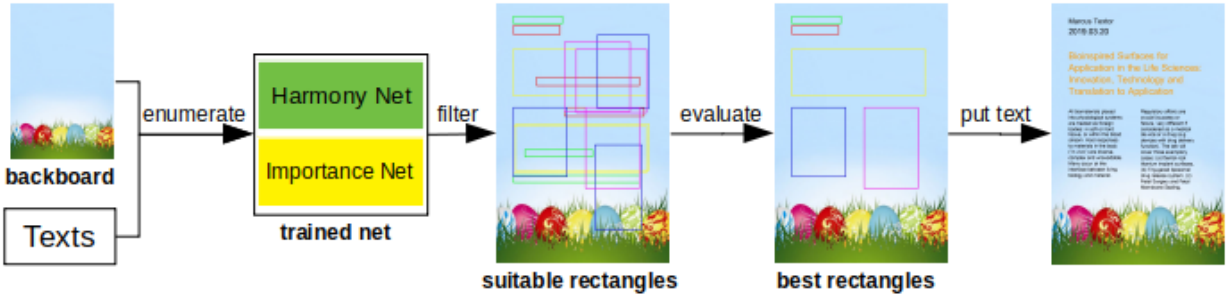


Figure 2: The pipeline of poster generator.

Text Type	Importance Level
title	4
date, presenter's name	3
detailed description of lecture	2
supplementary information	1

Table 1: The importance level of different text types.

a more prominent position on the poster. Therefore, we also need to know the importance of the position of any rectangular area on the background image. In our method, we employ a neural network to regress this attribute.

Data Preparation We collect some well-designed posters on the Internet. In these posters, the placement of each text is generally consistent with the importance of the text. Therefore, on each poster, we pick several rectangular blocks covered by text, extract their position and size information, and just take the corresponding importance level of the text as the importance of the rectangles. In addition, we also pick some rectangular blocks without text, and set their importance level to 0. Figure 1(b) shows such a process. After that, the importance level of all rectangles should be normalized from $0 \sim 4$ to $0 \sim 1$. Now, these data can be used to train our network.

Network This network is composed of three fully connected layers connected in sequence, and sigmoid function is used as the final activation function. The input format of this network is (x, y, w, h), where (x, y) is the relative position of the upper left corner of a rectangle on a poster, and w and h are its relative width and height compared with the whole poster. The output of the network is a value between 0 and 1. The larger the value, the higher importance of the input rectangular block.

In this problem, we employ mean square error as our loss function.

3.3 Poster Generator

In the generator, we select a best rectangle block for each t in T in descending order of importance of t. After we get two trained networks. We can calculate the importance and harmony of each rectangle. It can be used to find the rectangle that fits the given text. For $\forall t \in T$, we define $Rec(t)$ as the set of rectangles that fit the text t and we choose a best rectangle $R_t \in Rec(t)$ to place the text. For different text t_1, t_2, we should make sure that rectangles R_{t_1}, R_{t_2} don't intersect, which means that for $\forall t_1, t_2 \in T$ with $t_1 \neq t_2$, $R_{t_1} \cap R_{t_2} = \emptyset$. The pipeline of the poster generator is shown in Figure 2.

Enumerate rectangles to get the set Rec(t) For each text $t \in T$, we enumerate rectangles in the image and employ a filter to get $Rec(t)$. The filter is designed as: for each enumerated rectangle, we calculate its importance and harmony and record as (I, H). If (I, H) satisfies a bound (I_t, H_t) then the rectangle will be added to the set $Rec(t)$. For the enumeration step, We only consider rectangles that are parallel to the border of the image. Enumerate the width and top left corner of the rectangle. The height of the rectangle can be calculated in terms of width and the properties of the text(e.g., font, size, length, etc.). Note that we can't divide a word into two lines, so there may be some width that can't put a word on one line and we just ignore them.

Select the best rectangle in Rec(t) For each rectangle in $Rec(t)$, we first determine if it will intersect the rectangles selected for the previous texts. We can describe a rectangle as a quad (x_1, y_1, x_2, y_2), where (x_1, y_1) represents the upper left corner of the rectangle and (x_2, y_2) represents the lower right corner. Define a set $R = \{R_1, R_2, \ldots, R_n\}$ to represent the set of best rectangles selected for the previous texts. The rectangle we are checking now is R_{now}. If R_{now} can be used to place text, we should make sure that:

$$\forall R_i \in R, R_i \cap R_{now} = \emptyset \tag{3}$$

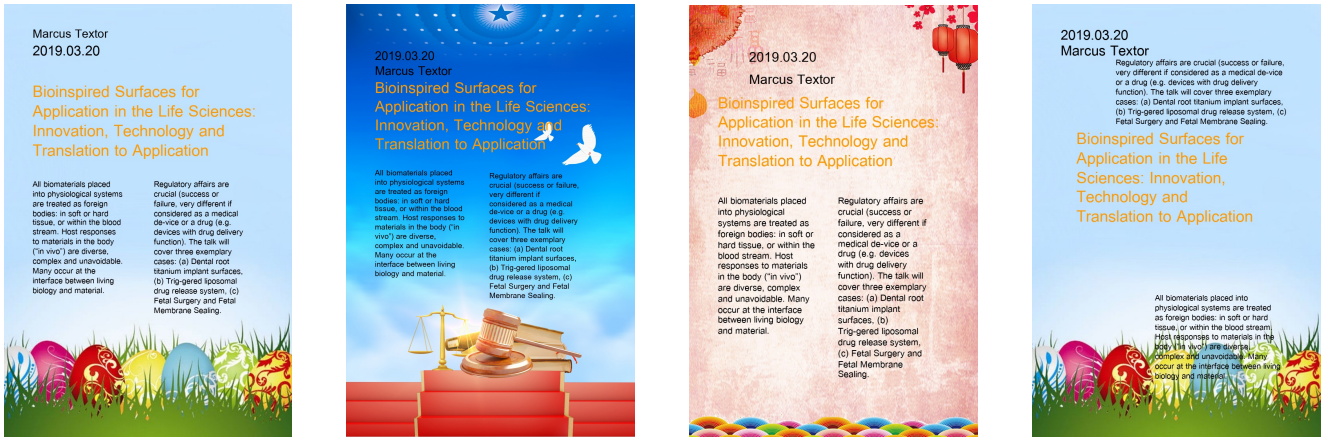
For any rectangle, we define four functions X_1, X_2, Y_1, Y_2 which denote x_1, y_1, x_2, y_2 of the rectangle respectively. For the two rectangles R_i and R_{now}, if at least one of the following holds then they don't intersect, otherwise, they intersect:

1. $X_2(R_i) < X_1(R_{now})$
2. $X_1(R_i) > X_2(R_{now})$
3. $Y_2(R_i) < Y_1(R_{now})$
4. $Y_1(R_i) > Y_2(R_{now})$

Evaluation function If R_{now} satisfies the condition in Equation 3, then we need to evaluate it. The normal evaluation function is:

$$h(I, H) = \alpha * I + \beta * H \tag{4}$$

where I, H are the importance and harmony of R_{now}, and α, β are two scale factors.In this function, we only consider the rectangle's importance and harmony. But for some different texts' rectangles, their positions are always related.



(a) (b) (c) (d)

Figure 3: Comparison of generated posters(a,b,c) and messy poster(d).

For example, different texts about detailed description of the lecture are usually aligned horizontally or vertically on one side. So we also adopt another function:

$$g(x_n, y_n, x_0, y_0) = \gamma * (|x_n - x_0| + |y_n - y_0|) \quad (5)$$

Where $x_n = X_1(R_{now})$, $y_n = Y_1(R_{now})$, γ is a scale factor and x_0, y_0 are the coordinates that R_{now} needs to be aligned. Finally, we combine these two functions as the final evaluation function:

$$\begin{aligned} &\Phi(I, H, x_n, y_n, x_0, y_0) \\ =&h(I, H) + g(x_n, y_n, x_0, y_0) \\ =&\alpha * I + \beta * H + \gamma * (|x_n - x_0| + |y_n - y_0|) \end{aligned} \quad (6)$$

If the rectangle doesn't need to align with other rectangles, we only need to set $x_n = x_1$ or $y_n = y_1$ or both of them. After the rectangles in $Rec(t)$ have been evaluated, we can select the best one from them for placing text t.

4. EXPERIMENT

4.1 Harmony between Image and Text

Dataset. We first collect some images that can be used as the background of posters. Then we select a fixed number of rectangular blocks on each poster, judge them artificially and generate labels. If a rectangular block is suitable for placing text, it's label is 1. Otherwise, its is 0.

Training Details. We divide one tenth of the dataset into validation dataset and the rest are training datset. For the k in the pooling layer's output size $3 \times k \times k$, it is finally set to 7 after many attempts and comparisons. Our classifier is trained using Adam, with a learning rate of 0.001.

Results. In this subtask, we compare our network with a baseline model CNN[3]. The final loss of our network on the validation test is 0.610, which is lower than the baseline 0.634. This shows that our network works better.

4.2 Importance of Position

Dataset. As mentioned in Section 3.2, we collect some well-designed posters on the Internet and pick several rectangles on each poster as data. When picking the rectangles without text, we make sure that these positions are suitable for text. Otherwise, the lack of text in these areas is most likely because the images in these areas are less harmonious, rather than the positions are not important enough.

Method	Ours	Linear-SVR	RBF-SVR	Bayesian Ridge
Loss	0.022	0.052	0.047	0.053

Table 2: The loss of different methods on importance of position.

Training Details. We divide one tenth of the dataset into validation dataset and the rest are training datset. This network is trained using Adam, with a learning rate of 0.001.

Results. In this subtask, we compare our network with several baselines including bayesian ridge regression[5] and support vector regression (SVR) with linear kernel and RBF kernel[8]. The final results are shown in Table 2. Our method has a better performance with loss 0.022, which is lower than all others.

4.3 Poster Generator

Data. The input data of the poster generator are some texts and an background image collected online.

Implementation details. Recall the evaluation function Φ. We set $\gamma \gg \alpha, \beta$, because all evaluated rectangles have satisfied the bound (I_t, H_t). In other words, if there are no other constraints, all rectangles $R_i \in Rec(t)$ can place text t. But the alignment is the new constraint and the more important one. So we will give a very large penalty for the rectangle which doesn't satisfy the alignment constraint well.

Result. We also design a simple generator without the help of the two networks or the guidelines we add in our method. There is a comparison of the posters generated by our method and the simple one in Figure 3. All the texts on the posters are placed in a neat place of the background which looks comfortable. The structure and organization of our posters are also very clear. And Figure 3(a,b,c) show that our generator can adapt to different backgrounds. To objectively evaluate our generator, we invite some people to score the generated lecture posters on reasonability and

aesthetics from 0 to 10. The result is that more than 80% of people give an average score higher than 7. This shows that our method is effective.

5. CONCLUSION

In this paper, we learned to generate lecture posters automatically from a background image and a text set. What we mainly need to consider was how to find an appropriate place for each piece of text. We designed two networks to determine the importance level of a place on the background and its harmony with text. Our generator generated lecture posters based on some design guidelines with the help of the two networks. The final results of the quality inspection experiments of the posters we generated showed the effectiveness of our method.

As the future work, our method can be generalized to generate other types of posters. It can also be applied to the work on putting text on images.

6. REFERENCES

[1] Cao, Y., Chan, A. B., and Lau, R. W. Automatic stylistic manga layout. *ACM Transactions on Graphics (TOG) 31*, 6 (2012), 141.

[2] Goodfellow, I., Pouget-Abadie, J., Mirza, M., Xu, B., Warde-Farley, D., Ozair, S., Courville, A., and Bengio, Y. Generative adversarial nets. In *Advances in Neural Information Processing Systems 27*, Z. Ghahramani, M. Welling, C. Cortes, N. D. Lawrence, and K. Q. Weinberger, Eds. Curran Associates, Inc., 2014, pp. 2672–2680.

[3] Krizhevsky, A., Sutskever, I., and Hinton, G. Imagenet classification with deep convolutional neural networks. *Advances in neural information processing systems 25*, 2 (2012).

[4] O'Donovan, P., Agarwala, A., and Hertzmann, A. Learning layouts for single-pagegraphic designs. *IEEE transactions on visualization and computer graphics 20*, 8 (2014), 1200–1213.

[5] Park, T., and Casella, G. The bayesian lasso. *Publications of the American Statistical Association 103*, 482, 681–686.

[6] Qiang, Y., Fu, Y., Guo, Y., Zhou, Z., and Sigal, L. Learning to generate posters of scientific papers. *CoRR abs/1604.01219* (2016).

[7] Qiang, Y.-T., Fu, Y.-W., Yu, X., Guo, Y.-W., Zhou, Z.-H., and Sigal, L. Learning to generate posters of scientific papers by probabilistic graphical models. *Journal of Computer Science and Technology 34*, 1 (Jan 2019), 155–169.

[8] Smola†, A. J., and Lkopf‡, B. S. A tutorial on support vector regression. *Statistics Computing 14*, 3 (2004), 199–222.

[9] Zhu, J.-Y., Park, T., Isola, P., and Efros, A. A. Unpaired image-to-image translation using cycle-consistent adversarial networks. In *The IEEE International Conference on Computer Vision (ICCV)* (Oct 2017).