GIF Super-Resolution

Yizhou Wang
Columbia University
wang.yizhou@columbia.edu

Liangliang Cao
Hellovera AI
liangliang.cao@gmail.com

Abstract

Although GIFs have been popular on the Internet, they suffer from slow download speed due to the reliance on its old coding schemes and thus lead to large file sizes. To solve this problem, this paper proposes a novel super-resolution approach for GIFs, which uses two high-resolution frames (the first and last frames) as well as the low-resolution data to generate a high-resolution GIF. The two single frames are very small compared with the size of the GIFs, so that our approach requires little extra cost but brings impressive performance boost. This approach is motivated by the unique characteristics of GIFs, especially GIFs are usually much shorter than videos but have quite diverse and dynamic content. To validate this approach, we collect a new super-resolution dataset for GIFs, which is the first dataset devoted to GIF super-resolution, while containing significantly more sequences than previous video super-resolution datasets. The experiments on this dataset show that the performance of our algorithm significantly outperforms the popular video super-resolution baselines while achieving at least 80 *times speedup on CPU.*

1. Introduction

The format of GIF (Graphics Interchange Format) was first introduced in 1987, but only gained their popularity over the last few years. The reason for its popularity is because GIFs facilitate mobile communication and encourage Web engagement. GIFs are more attractive than still images and easier to display and share than traditional videos. According to Twitter[1], people shared more than 100 million GIFs in 2015 through tweets.

Despite its popularity, GIF is an old format created 30 years ago and the compression algorithm for GIF is far from being efficient. The format supports up to 8 bits per pixel for each image, which supports only up to 256 different colors, which are significantly reduced compared to the popular 24-bit RGB color space. These limitations make GIF less effective in representing color photographs and color gradients. GIF images are based on lossless compression, which uses the LempelZivWelch (LZW) algorithm to reduce the file size. Due to the nature of the lossless compressing algorithm, GIF files are often much bigger than their counterparts in videos of the formats of MP4 or MOV. In practice, GIFs are usually encoded with low spatial resolutions.

This paper aims to address the problem of limited resolutions associated with animated GIFs. The idea of super-resolution is to first transfer a low-resolution GIF and then decode it to the high-resolution on the client side. Such an approach has been studied in similar fields, especially image super-resolution and video super-resolution.

In practice the GIF super-resolution is very different from videos due to the following characteristics: (1) GIFs involve no audio, (2) GIFs are usually very short in duration, and (3) when the bandwidth is not high enough (especially on mobile devices), users first see only the first frame instead of the animated GIFs unless the user clicks to download.

To study the problem of GIF super-resolution, we collect a large dataset with various GIF categories. It contains 1134 GIFs including emotion, action, animation, scene, and animal. Compared with the popular video datasets with dozens to hundreds of videos developed for super-resolution research, our dataset is significantly larger. To the best of our knowledge, this is the first dataset for GIF super-resolution. We will release this dataset for future research.

Based on the new dataset, we develop a simple but efficient approach for GIF super-resolution as shown in Figure 1. We propose to augment the low-resolution GIF with two high-resolution images corresponding to the first and last frames. The cost of introducing two high-resolution images is very low: two frames are small compared with the size of the whole GIF, and they can be shown as the still illustration before loading the entire animated GIF on mobile devices. However, our approach leads to significantly better GIF super-resolution results. The peak signal-to-noise ratio (PSNR) of our approach is consistently higher than the state-of-the-art video super-resolution approach, while our approach is at least 80 times faster. Moreover, our method

[1]https://blog.twitter.com/official/en_us/a/2016/introducing-gif-search-on-twitter.html

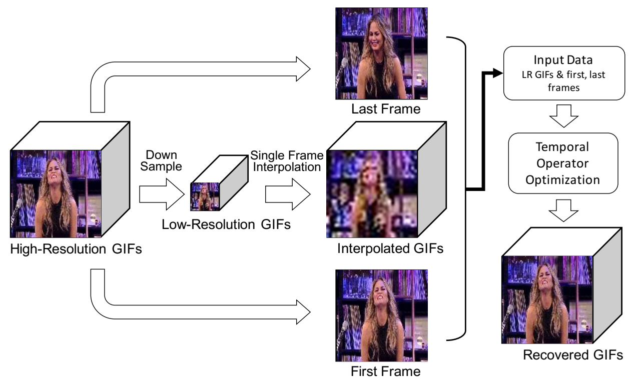


Figure 1: Diagram of our GIF SR method.

can compress the file size by 70%, which means that user only needs to download 30% size of a high-resolution GIF to review the GIF on mobile devices. We believe our approach is very promising and encourage more effective approaches in the future.

The contribution of this work is three-fold: (1) it collects a new large GIF super-resolution dataset and release it as a public benchmark; (2) it proposes to expand the traditional GIF data by adding two high-resolution frames (the first and last frames); (3) it develops a simple but effective super-resolution approach exploiting the proposed expanded format which outperforms the popular baseline for video super-resolution while achieving at least 80 times speedup on CPU.

This paper is organized as follows: Firstly, we explored some research works related to image super-resolution and video super-resolution in Section 2. Then, in Section 3, we collected the first GIF super-resolution dataset – GIFSR. After that, we proposed a GIF super-resolution method in Section 4, which combining Bicubic Interpolation and a proposed temporal operator. The experiments of our method are performed on GIFSR dataset in Section 5 and we conclude the paper in Section 6.

2. Related Work

There are mainly two groups of works on super-resolution: image super-resolution and video super-resolution. A large proportion of the existing studies [9][6][10][5] are devoted to image super-resolution. However, these studies do not consider the motion information and multiple frame scenarios, and hence are not good candidates for our problem.

Video super-resolution extends the techniques in image super-resolution to generate high-resolution frames. A number of recent studies [11][12][7][8][13] are carried out by combining single frame appearance and motion information. Especially, Kappeler *et al*. [8] proposed VSRnet, which trains CNN based network on both the spatial and the temporal dimensions to enhance their spatial resolution, and obtains the state-of-the-art results for video super-resolution.

One limitation of existing video super-resolution methods lies in the lack of large-scale video database for training. Although there are a number of large-scale image database including ImageNet, MIT Scene, and YFCC100M, the datasets used for video super-resolution are much smaller. For example, *Videoset4* [4] has only 4 videos *Myanmar* [3] has 59 videos, and *CDVL* [1] has 115 videos. Due to the dataset limitation, some video super-resolution studies [8] have to use from ImageNet to pre-trained the model, and adapt the model to video data. Since many recent works suggest large-scale data will help to improve the performance, we collect a new super-resolution dataset with more than 1000 GIF sequences and hope it will benefit the future research in super-resolution.

Another limitation of many existing video super-resolution algorithms is a slow speed. For example, it may take VSRnet [8] about 20 seconds to several minutes to process a short video sequence. Such a speed is not acceptable for Internet users who want to view or share a GIF with their friends. To solve this problem, this paper focuses on efficient solutions. As a result, our new algorithm can be 80 times faster than [8] for GIF super-resolution.

3. Dataset

In this work, we collect the first large GIF super-resolution dataset for our study. The GIFs in the dataset are collected from *GIPHY* Website [2]. We collected 1134 GIFs and grouped into five categories: emotion, action, scene, animation, and animal. In the following sections, we will name our dataset as "GIFSR" where "SR" stands for the abbreviation of super-resolution.

Some sample GIFs in GIFSR dataset are shown in Figure 2. The category distribution of the GIFs and their average length (frame number) are shown in Table 1. The category that contains most GIFs is "emotion set", which will be used in many of the following experiments in Section 5.

The GIFs are with various lengths from 6 to 400 frames and the distribution of the frame numbers are shown in Figure 3. From the figure, we can find out that most of the GIFs are distributed in the range of $[6, 50]$. This is because the GIFs that we commonly use are short in length and do not contain rich information.

Then, we compare our dataset with other datasets for video super-resolution. As shown in Table 2, our dataset is significantly larger than the previous ones.

(a) Emotion

(b) Action

(c) Scene

(d) Animation

(e) Animal GIFs.

Figure 2: Sample GIFs in GIFSR dataset by categories.

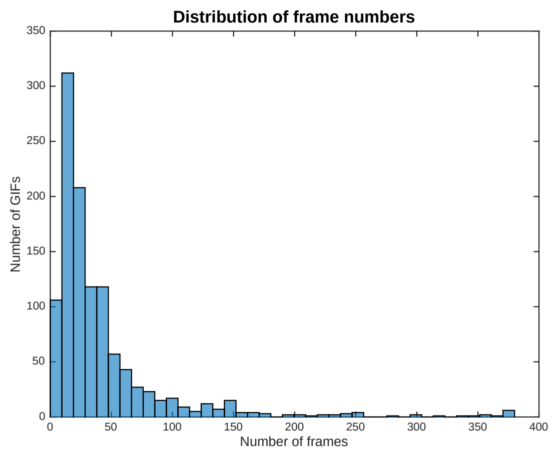


Figure 3: Distribution of frame numbers in GIFSR dataset.

Category	Nr. of GIFs	Avg. frame
Emotion	446	35.6
Action	256	39.0
Scene	123	64.5
Animation	266	40.1
Animal	43	80.2
Total	1134	42.24

Table 1: Category distribution and the average frame in GIFSR dataset.

Dataset	Nr. of items	Resolution
Myanmar[3]	59	960×540
Videoset4[4]	4	$702 \times 576^*$
CDVL[1]	115	1920×1080
GIFSR	1176	128×128

* Vid4 is composed of walk, city, calendar and foliage, and has sizes 720×480 or 720×576.

Table 2: Comparison between GIFSR and traditional video super-resolution sets.

4. Our Method

Based on these unique characteristics, our super-resolution method takes the low-resolution GIF together with two high-resolution images corresponding to the first and last frames. The cost of introducing two high-resolution images is very low: two frames are small compared with the size of the whole GIF, and can be shown as the still illustration before loading on mobile devices. However, the benefit of introducing two high-resolution images leads to a novel solution with nice super-resolution performance. Following this simple idea, we assumed the first frame and the last frame of a GIF is known. Our model is to use the low-resolution GIF as well as the high-resolution first and last frames to predict the high-resolution GIF.

To model the temporal relationship between GIF frames, we proposed a temporal operator to predict the current frame using the past frame and the low-resolution version of the current frame. The forward operation between each frame is

$$F_i = \rho_f F_{i-1} + \gamma_f f_i,$$

and the backward operation is

$$F_i = \rho_b F_{i+1} + \gamma_b f_i,$$

where F_i is the iteration result of the i-th frame, and f_i is the Bicubic Interpolation (BI) result of the i-th frame. ρ_f, γ_f, ρ_b, γ_b are coefficients of the temporal operator.

So the last frame F_n can be computed using the ground truth of the first frame F_0^{GT} and the BI of the middle frames

$\{f_i\}_{i\in[1,n]}$.

$$\begin{aligned} F_n &= \rho_f F_{n-1} + \gamma_f f_n \\ &= \rho_f(\rho_f F_{n-2} + \gamma_f f_{n-1}) + \gamma_f f_n \\ &= \cdots \\ &= \rho_f^n F_0^{GT} + \sum_{i=1}^{n} \rho_f^{n-i} \gamma_f f_i. \end{aligned}$$

Then, we do an optimization using the ground truth of the last frame F_n^{GT},

$$\min_{\rho_f,\gamma_f} l_f(\rho_f,\gamma_f) = \min_{\rho_f,\gamma_f} \left\| F_n - F_n^{GT} \right\|^2.$$

To solve this optimization problem, use gradient descent on the object function $l_f(\rho_f,\gamma_f)$. The partials of $l_f(\rho_f,\gamma_f)$ respected to coefficients ρ_f and γ_f are

$$\begin{cases} \dfrac{\partial l_f(\rho_f,\gamma_f)}{\partial \rho_f} = 2F_n \dfrac{\partial F_n}{\partial \rho_f} - 2F_n^{GT} \dfrac{\partial F_n}{\partial \rho_f}, \\ \dfrac{\partial l_f(\rho_f,\gamma_f)}{\partial \gamma_f} = 2F_n \dfrac{\partial F_n}{\partial \gamma_f} - 2F_n^{GT} \dfrac{\partial F_n}{\partial \gamma_f}, \end{cases}$$

where

$$\begin{cases} \dfrac{\partial F_n}{\partial \rho_f} = nF_0^{GT}\rho_f^{n-1} + \displaystyle\sum_{i=1}^{n-1}(n-i)\rho_f^{n-i-1}\gamma_f f_i, \\ \dfrac{\partial F_n}{\partial \gamma_f} = \displaystyle\sum_{i=1}^{n} \rho_f^{n-i}\gamma_f f_i. \end{cases}$$

Thus, the gradient of $l_f(\rho_f,\gamma_f)$ is

$$\nabla l_f = \left(\frac{\partial F_n}{\partial \rho_f}, \frac{\partial F_n}{\partial \gamma_f} \right).$$

Similarly, we also have an optimization using the ground truth of the first frame F_0^{GT},

$$\min_{\rho_b,\gamma_b} l_b(\rho_b,\gamma_b) = \min_{\rho_b,\gamma_b} \left\| F_0 - F_0^{GT} \right\|^2.$$

The gradient of $l_b(\rho_b,\gamma_b)$ is

$$\nabla l_b = \left(\frac{\partial F_0}{\partial \rho_b}, \frac{\partial F_0}{\partial \gamma_b} \right).$$

Note $\{F_i\}_{i\in[0,n]}$ as the results of super-resolution, $\{f_i\}_{i\in[0,n]}$ as the BI of the i-th frame. We consider the cost function

$$l(\rho_f,\gamma_f,\rho_b,\gamma_b) = ||F_n - F_n^{GT}||^2 + ||F_0 - F_0^{GT}||^2, \quad (1)$$

where the super-resolution result is computed by

$$F_i = \rho_f F_{i-1} + \gamma_f f_i.$$

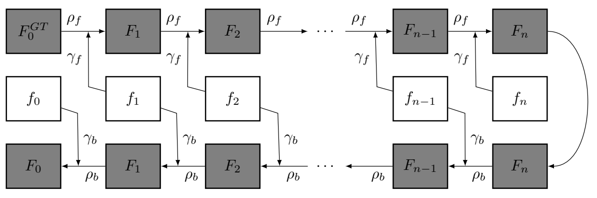


Figure 4: Block diagram of our model.

The block figure of the above model is shown in Figure 4.

In practice, we can choose the first half of the cost function (1), whose gradient can be computed efficiently

$$\begin{cases} \dfrac{\partial l(\rho_f,\gamma_f)}{\partial \rho_f} = 2F_n \dfrac{\partial F_n}{\partial \rho_f} - 2F_n^{GT} \dfrac{\partial F_n}{\partial \rho_f}, \\ \dfrac{\partial l(\rho_f,\gamma_f)}{\partial \gamma_f} = 2F_n \dfrac{\partial F_n}{\partial \gamma_f} - 2F_n^{GT} \dfrac{\partial F_n}{\partial \gamma_f}. \end{cases}$$

Put them together, GIF Super-Resolution algorithm is proposed as Algorithm 1.

Algorithm 1: GIF-SuperResolution

```
Input  : Low-resolution GIF, High-resolution first
         frame $F_0^{GT}$ and last frame $F_n^{GT}$.
Output: High-resolution GIF result.
Extract frames $F_i^{LR}$ from the low-resolution GIF, the
  number of frame is $n$;
for $i \leftarrow 0$ to $n$ do
    $f_i \leftarrow$ BicubicInterp($F_i^{LR}$);
end
Initialize parameters $\rho_f^{(0)}, \gamma_f^{(0)}$;
Gradient descent parameters: step size $\epsilon$, number of
  iterations $t$;
$\rho_f^{(t)}, \gamma_f^{(t)} \leftarrow$
  GradientDescent($\{f_i\}_{i\in[0,n]}, F_0^{GT}, F_n^{GT}$);
for $i \leftarrow 1$ to $n$ do
    $F_i = \rho_f^{(t)} F_{i-1} + \gamma_f^{(t)} f_i$;
end
Merge the frames $F_0^{GT}, \{F_i\}_{i\in[1,n-1]}, F_n^{GT}$ to GIF;
```

5. Experiment

In this section, we first applied our method of GIF super-resolution on GIFSR dataset. Then, we compared our PSNR results with two baselines Bicubic Interpolation and VSRnet [8]. After that, we analyzed the execution time for our method to illuminate that our algorithm can be really fast in the application. Moreover, we analyzed the file size

Category	BI PSNR	Our results
Emotion	20.25	23.91
Action	18.90	21.48
Scene	18.07	20.97
Animation	18.38	21.93
Animal	20.10	22.68
Total	19.27	22.53

Table 3: The results for PSNR results of different categories in GIFSR dataset with resolution combination of (HR, LR) $= (128, 32)$.

of the input and output of our method to show the potentials of our algorithm for real-world GIF data.

Our experiment for Bicubic Interpolation was performed on a MacBook Pro with 2.7 GHz Intel Core i5 processor and experiment for VSRnet was performed on a Linux workstation with an NVIDIA GeForce GTX 980 graphics card.

5.1. PSNR of Our Method

We implemented our proposed method with Python. The high-resolution (HR) of our dataset is 128×128 pixels and the low-resolution (LR) GIF to be processed is 32×32 pixels. Here, denote (HR, LR) as resolution combination. During the gradient descent, we set the step size to be 10^{-8} and number of iterations to be 50. Before iteration, we initialize $\rho_f = \rho_b = [0.9]_{128\times128\times3}$, $\gamma_f = \gamma_b = [0.1]_{128\times128\times3}$.

After implemented our method on GIFSR dataset, we got an average PSNR of 22.53. Some example result frames are shown in Figure 5. The PSNR of each category is shown in Table 3. From the results, we can conclude that the PSNR of our method is better than BI in all five categories in GIFSR. Among all the five categories, emotion set obtains the greatest improvement using our method. A proper reason could be that the average length of emotion set is shortest; the motion of the emotion set is the weakest so that it would be easier to do super-resolution.

We also did some experiments of VSRnet [8] on GIFSR emotion set and obtained the following PSNR results shown in Table 4. Here, we down-sampled the GIFs in GIFSR dataset to three different resolution combinations (HR, LR) $= (32, 8), (64, 16), (128, 32)$. There is a technique when implementing VSRnet. Since VSRnet only supports grayscale frames, we input the three channels of each frame separately and combined the result for three channels at the end.

Based on the experiments, we summarize our observation as the following: (1) The PSNR of lower resolution combinations is less than higher resolution combinations in all three methods. (2) The performance of VSRnet is bad when resolution combinations are low, even worse than

Model	(HR, LR)	PSNR
Bicubic	$(32, 8)$	15.66
	$(64, 16)$	17.90
	$(128, 32)$	20.25
VSRnet	$(32, 8)$	13.42
	$(64, 16)$	18.48
	$(128, 32)$	22.10
Our result	$(32, 8)$	22.28
	$(64, 16)$	23.17
	$(128, 32)$	23.91

Table 4: Comparison between PSNR of Bicubic Interpolation, VSRnet and our method on GIFSR emotion set with different resolution combinations.

Bicubic Interpolation. (3) The PSNR of our method is better than other two methods in every condition.

As mentioned in Section 3, there are more GIFs with small frame number in the range of $[6, 50]$ in GIFSR dataset. Thus, the frame number should not be a constraint in GIF super-resolution algorithms. However, we want to know deeper about the effect of frame number in our method. So that we plot some scatters as Figure 6. Each point in the scatters represents a GIF item and the color represents its category shown in the legend. The y-axis "PSNR increment" is the increment of PSNR from Bicubic Interpolation to our results. In Figure 7b, the points are intensive in the middle-left area. Though it is not easy to find out the relationship, there are few points in the upper-right area, which means the super-resolution for long GIFs are difficult to be improved by our method. To make it clear, we changed the x-axis to $\log(\text{frame number})$ and performed linear regression on the data points. The linear regression result is shown in Figure 6b. The slope of the black line is -0.69 which shows the effect of our method drops while the frame number climbs. But this effect is not significant so that our method is also applicable to long GIFs.

5.2. Execution Time Analysis

Since GIFs are usually used on Web or mobile devices, the execution time of GIF super-resolution algorithm should be real-time so that people do not need to wait for processing or loading the GIFs.

Thus, we record the execution time of three methods on GIFSR emotion set. In Table 5, we listed these execution time of different resolution combinations. Bicubic interpolation is the fastest algorithm with speed of approximately 0.04 second per frame. VSRnet processed really slow even with a low resolution combination $(32, 8)$. Our method, however, gave a good enough execution time on CPU, which could be at least 80 times faster than VSR-

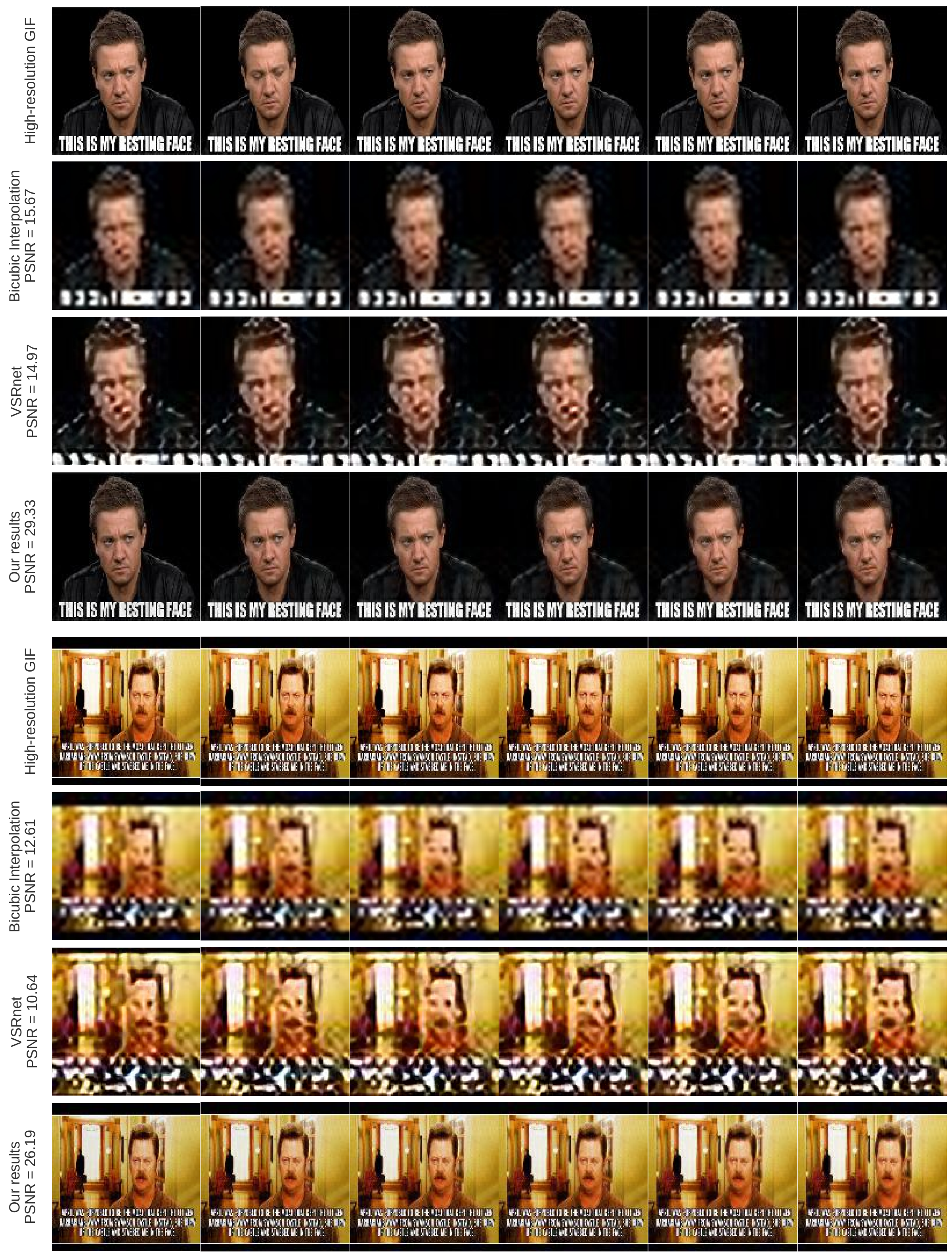


Figure 5: Sample results for our method on GIFSR dataset.

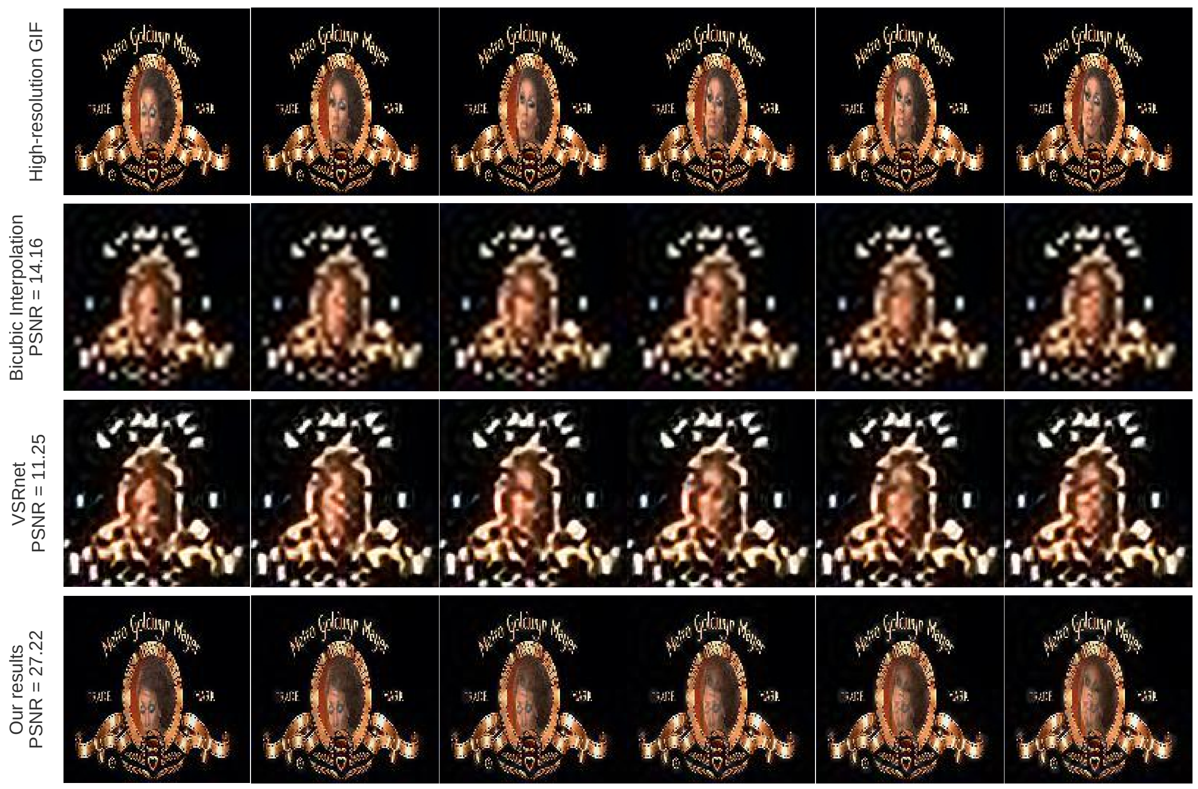


Figure 5: Sample results for our method on GIFSR dataset. (Cont'd.)

Model	(HR, LR)	Per-frame Time	Per-GIF Time
Bicubic*	$(32, 8)$	0.0373	1.3267
	$(64, 16)$	0.0395	1.4041
	$(128, 32)$	0.0538	1.9128
VSRnet	$(32, 8)$	7.3692	262.0845
	$(64, 16)$	12.3024	437.5392
	$(128, 32)$	20.4888	728.6823
Our method	$(32, 8)$	0.0676	2.4041
	$(64, 16)$	0.1178	4.1905
	$(128, 32)$	0.3088	10.9813

* The execution time of Bicubic Interpolation includes frame extraction and combination processes.

Table 5: Comparison between execution time (in seconds) of Bicubic Interpolation, VSRnet and our method on GIFSR emotion set with different resolution combinations.

net on GPU. With a speed of 0.3 second per frame on a $(128, 32)$ GIF, this algorithm can be treated as real-time and potentially can apply on Web or mobile devices.

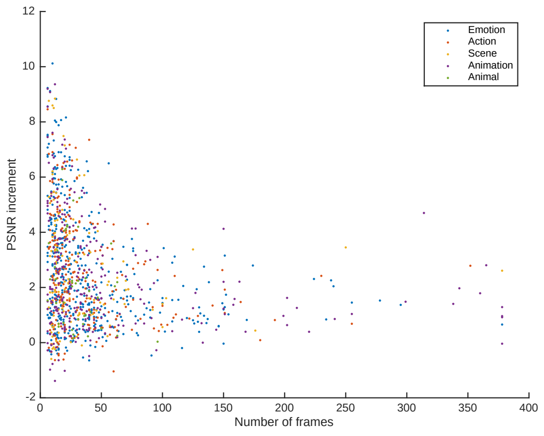
Moreover, another factor that can affect execution time is the number of iterations during gradient descent. Therefore, we changed the number of iterations to be $\tau = \{10, 20, 30, 50, 100\}$ and compare their PSNR results and execution times. The results are shown in Figure 7. The PSNR is improved when the number of iterations increases and execution time grows linearly with the number of iterations.

5.3. File Size Analysis

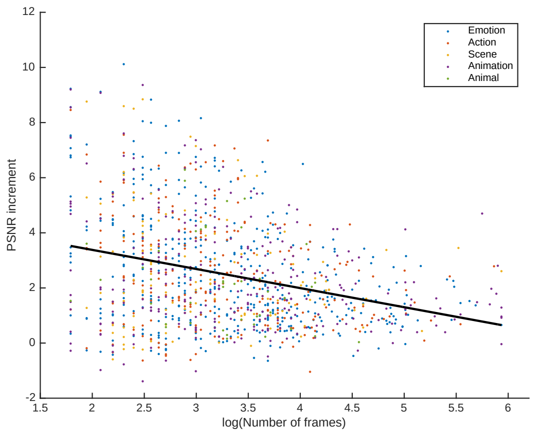
Finally, to demonstrate our algorithm can be used to compress GIF, we summarized the file size of high-resolution GIFs, which are the files that should be downloaded by users, and low-resolution frames plus high-resolution first and last frames, which are the input of our GIF super-resolution algorithm in Table 6. Here, we listed the size of high-resolution frames in the table and it only accounts for a small proportion (around 15%) of the size of all input files. Looking at the file size of the high-resolution GIFs and input files, we can conclude that, using our method, a GIF with a resolution of 128×128 pixels can be compressed by 70%.

6. Conclusion

In this paper, we established a GIF super-resolution dataset "GIFSR", containing 1134 items divided into five



(a) Distribution of the frame number and PSNR results.



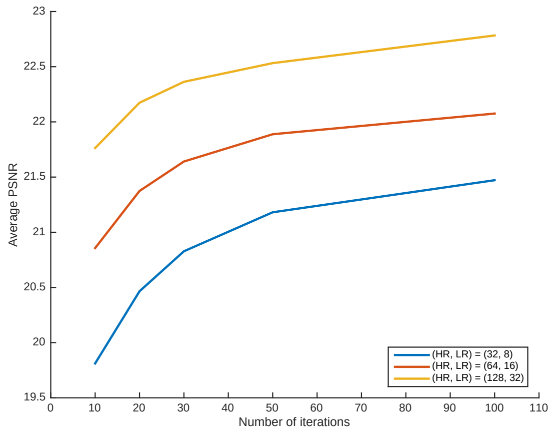
(b) Distribution of the logarithm of frame number and PSNR results. The black line is the linear regression of the data points.

Figure 6: The relationship between PSNR results and frame number.

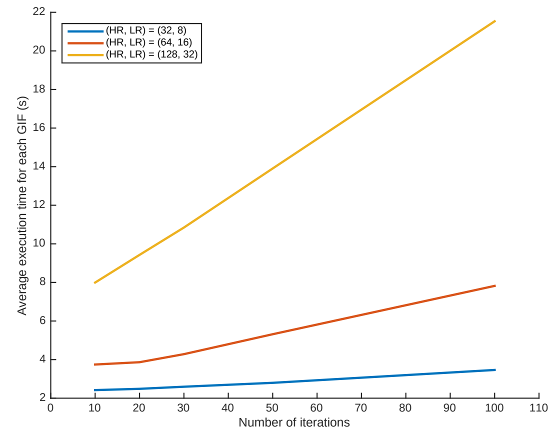
(HR, LR)	HR GIF	Input files	Compressed
$(32, 8)$	29.6MB	$0.8/11.4$MB	18.2MB
$(64, 16)$	77.9MB	$1.7/13.5$MB	64.4MB
$(128, 32)$	235.8MB	$3.5/19.5$MB	216.3MB

Table 6: Comparison between file size of high-resolution GIF and the input of our method on GIFSR emotion set. Here input files contain LR frames and HR first and last frames and data is in the format of “size of HR frames / size of all input files”.

categories. Then, we proposed a GIF super-resolution method combining bicubic interpolation and temporal operator. The proposed method was tested on GIFSR dataset and obtained higher PSNR than bicubic Interpolation and VSRnet baselines. Moreover, we analyzed the execution time of our method on CPU which outperformed at least 80 times than VSRnet on GPU. Finally, the file size of high-resolution and low-resolution GIFs are collected and we concluded that GIFs can be compressed by 70% using our method.



(a) Average PSNR results.



(b) Average execution time.

Figure 7: The effect of the number of iterations in gradient descent on PSNR results and execution times for different resolution combinations on GIFSR.

7. Acknowledgement

We are very thankful to Shih-Fu Chang for his insightful suggestions and comments. L. Cao would like to thank Ding Liu for the discussions on video super-resolution.

References

[1] The consumer digital video library. Institute for Telecommunication Sciences. Available: `http://www.cdvl.org/`. Online; Accessed: 2017-11-25.

[2] Giphy: Search all the gifs & make your own animated gif. Available: `https://giphy.com/`. Online; Accessed: 2017-11-25.

[3] Myanmar 60p. Harmonic Inc. (2014). Available: `http://www.harmonicinc.com/resources/videos/4k-video-clip-center`. Online; Accessed: 2017-11-25.

[4] Videoset4. Available: `https://twitter.app.box.com/v/vespcn-vid4`. Online; Accessed: 2017-11-25.

[5] R. Dahl, M. Norouzi, and J. Shlens. Pixel recursive super resolution. *arXiv preprint arXiv:1702.00783*, 2017.

[6] K. Hayat. Super-resolution via deep learning. *arXiv preprint arXiv:1706.09077*, 2017.

[7] E. Ilg, N. Mayer, T. Saikia, M. Keuper, A. Dosovitskiy, and T. Brox. Flownet 2.0: Evolution of optical flow estimation with deep networks. *arXiv preprint arXiv:1612.01925*, 2016.

[8] A. Kappeler, S. Yoo, Q. Dai, and A. K. Katsaggelos. Video super-resolution with convolutional neural networks. *IEEE Transactions on Computational Imaging*, 2(2):109–122, 2016.

[9] R. Keys. Cubic convolution interpolation for digital image processing. *IEEE transactions on acoustics, speech, and signal processing*, 29(6):1153–1160, 1981.

[10] J. Kim, J. Kwon Lee, and K. Mu Lee. Accurate image super-resolution using very deep convolutional networks. In *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition*, pages 1646–1654, 2016.

[11] C. Liu and D. Sun. On bayesian adaptive video super resolution. *IEEE transactions on pattern analysis and machine intelligence*, 36(2):346–360, 2014.

[12] O. Makansi, E. Ilg, and T. Brox. End-to-end learning of video super-resolution with motion compensation. In *German Conference on Pattern Recognition*, pages 203–214. Springer, 2017.

[13] W. Shi, J. Caballero, F. Huszár, J. Totz, A. P. Aitken, R. Bishop, D. Rueckert, and Z. Wang. Real-time single image and video super-resolution using an efficient sub-pixel convolutional neural network. *IEEE Conference on Computer Vision and Pattern Recognition*, 2016.