AugLy: Data Augmentations for Robustness

Zoë Papakipos and Joanna Bitton
Meta AI
{zoep, jbitton}@fb.com

Abstract

We introduce AugLy, a data augmentation library with a focus on adversarial robustness. AugLy provides a wide array of augmentations for multiple modalities (audio, image, text, & video). These augmentations were inspired by those that real users perform on social media platforms, some of which were not already supported by existing data augmentation libraries. AugLy can be used for any purpose where data augmentations are useful, but it is particularly well-suited for evaluating robustness and systematically generating adversarial attacks. In this paper we present how AugLy works, benchmark it compared against existing libraries, and use it to evaluate the robustness of various state-of-the-art models to showcase AugLy’s utility. The AugLy repository can be found at https://github.com/facebookresearch/AugLy

1. Introduction

Data augmentations are a key component in the computer vision model development life cycle[24], and are also becoming increasingly prevalent in other domains[7]. They are commonly used to increase the size of datasets and prevent overfitting by performing perturbations on the input data. In addition to the classical use cases, data augmentations can also be used to evaluate the robustness of trained models to perturbations not seen at train time[11][10].

For instance, to preserve a sense of data provenance, being robust to data manipulations is critical. Content online is often manipulated and reshared, for example when users screenshot & share a post, or overlay text or images on top of an image to make a meme. It is therefore non-trivial to be able to detect that two pieces of media are near-duplicates [17]. Additionally, adversaries may try to intentionally pass in obfuscated data to a model to evade detection.

The classical set of data augmentations used during model development does not completely mimic the way individuals online organically perturb data. Most classical augmentation libraries focus on simple transformations such as mirroring, rotating, cropping, brightness changes, etc. While these kinds of augmentations do naturally occur online, others such as overlaying text and emojis, social media screenshots, etc. are also prevalent. In addition, multimodal data processing and learning is becoming increasingly important as many real-world use cases involve multiple types of data, such as text & images or audio & video,
and it can be useful to augment data of multiple modalities under one unified library & API.

AugLy is built with robustness and the vast landscape of organic data augmentations seen online in mind, and to our knowledge is the first multimodal data augmentation library. AugLy can be used to synthetically create realistic data augmentations seen online, as a tool for evaluating and increasing robustness to and augment multiple modalities at a time, and thus stands out in comparison to existing libraries. In this paper we introduce AugLy, explain how it works, its architecture, and how it compares in terms of functionality & efficiency to existing data augmentation libraries. We also conduct a robustness evaluation on state-of-the-art image classification models throughout the years to demonstrate how AugLy can be used to identify robustness gaps in pretrained models.

2. Related Work

Most commonly-used augmentation libraries focus on one modality and provide a fairly limited set of augmentations. A majority of libraries focus on images [1, 19, 14] and text [18, 8, 21], however audio [20, 29, 15] and video [32, 6, 16] augmentation libraries do exist as well with more limited augmentations (see Section 4 for in-depth comparisons between AugLy and existing libraries for each modality). Meanwhile, AugLy provides augmentations for audio, images, text and video under a unified API, and is one of few libraries[8] that focus on evaluating robustness rather than augmenting a dataset at train time.

Other works have conducted experiments to find sets of augmentations that when trained on improve robustness at test time, such as AugMix[12]. Strategies like AutoAugment, on the other hand, find an “optimal” set of augmentations to train on in a more automated way[2].

In AI Fairness, studies assessing the robustness of models to various protected categories are common. In NLP, there are studies that augment text to assess a model’s biases towards gender [31, 3] and ethnicity[26]. AugLy provides “fairness augmentations” since being robust to perturbations in protected classes is an important aspect of robustness that we must evaluate to ensure that models are not amplifying biases.

3. AugLy

AugLy is a novel open-source data augmentation library which provides over 100 data augmentations across four modalities: audio, image, text, and video. The augmentations provided in AugLy are informed by the perturbations that real people on the Internet perform on data daily. This includes augmentations such as overlaying text, emojis, and screenshot transforms for image & video and inserting punctuation or similar characters for text.
```python
import augly.video as vidaugs
import augly.audio as audaugs

aud_augs = audaugs.Compose(
    [
        audaugs.AddBackgroundNoise(),
        audaugs.Tempo(factor=2.0),
    ],
)

vid_augs = vidaugs.Compose(
    [
        vidaugs.Rotate(p=0.5),
        vidaugs.TimeCrop(
            offset_factor=0.2,
            duration_factor=0.4,
        ),
        vidaugs.AugmentAudio(
            audio_aug_function=aud_augs,
        ),
    ],
)

vid_augs(vid_path, out_path)
```

Figure 4. Composing audio & video augmentations

3.2. Existing Use Cases

AugLy has already been used by several projects. SimSearchNet[27], an image copy detection model, was trained using AugLy augmentations. AugLy was used to evaluate the robustness of deepfake detection models in the 2019 Deepfake Detection Challenge[4], ultimately influencing who were the top five winners. The dataset (DISC21) for the Image Similarity Challenge[5], a NeurIPS 2021 competition on image copy detection, was built using AugLy as well.

4. Benchmarking

In order to show how AugLy fits into the existing ecosystem of data augmentation libraries, we compare each modality’s sub-library within AugLy to a few of the most popular augmentations libraries in that respective modality. Specifically, we compare the overall focus and functionality of each library, and perform runtime benchmarking to evaluate how efficient AugLy’s augmentations are. Note: the augmentations were benchmarked using AugLy v0.2.1, available on Pypi and GitHub. To see the full list of augmentations benchmarked, please review the Appendix.

4.1. Audio

We chose to compare AugLy’s audio augmentations to three existing and popular libraries: pydub[23], torchaudio[29], and audiomentations[13]. See Figure 5 to compare the number of distinct augmentations provided.

Each library has a slightly different focus: torchaudio and audiomentations integrate easily with pytorch (torchudio’s can also be GPU-accelerated) and are clearly intended to be used at train time to improve generalization of audio machine learning models. Pydub provides more general-purpose audio processing functionality without much emphasis on either integrating with ML training or evaluation pipelines; the number of transformation functions in Pydub is also much lower than the other three.

We benchmark each audio augmentation in AugLy, as well as some analogues that exist in the other libraries. See Figure 6 for the runtime in seconds of each augmentation in (1) AugLy, (2) pydub, (3) torchaudio, & (4) audiomentations.

![Figure 5](image_url)  
**Figure 5.** The audio augmentation libraries we chose to compare and their corresponding number of augmentations at the time of writing.

| Library            | # augmentations |
|--------------------|-----------------|
| pydub              | 10              |
| AugLy              | 20              |
| audiomentations    | 25              |
| torchaudio         | 58              |

![Figure 6](image_url)  
**Figure 6.** The runtime (in seconds) of audio augmentations in (1) AugLy, (2) pydub, (3) torchaudio, & (4) audiomentations.

4.2. Image

We compare AugLy’s image augmentations to three well-established libraries: imgaug[14], torchvision[19], and Albumentations[1]. See Figure 7 for a comparison of the four libraries in terms of the number of distinct augmentations provided.

![Figure 7](image_url)  
**Figure 7.** The runtime (in seconds) of image augmentations in (1) AugLy, (2) pydub, (3) torchaudio, & (4) audiomentations.
4.3. Text

We compare AugLy’s text augmentations to three existing text libraries: nlpaug[18], TextAttack[21], & textflint[8]. See Figure 9 for a comparison of the five libraries in terms of the number of distinct augmentations provided.

One significant difference between AugLy and the other text augmentation libraries is the prevalence of syntactic versus semantic (i.e. character-level vs word-level) augmentations. Most augmentations in nlpaug and TextAttack are semantic (e.g. words being swapped for synonyms or antonyms), or a few simple syntactic ones (e.g. deleting/adding characters, replacing characters with nearby ones on the keyboard). AugLy provides many syntactic augmentations that are often used online in an attempt to evade detection, such as inserting punctuation, zero-width, or bidirectional characters and changing fonts.

We benchmark each AugLy text augmentation, as well as any analogues that exist in the other libraries. See Figure 10 for the runtime in seconds of each augmentation in (1) AugLy, (2) nlpaug, (3) TextAttack, & (4) textflint.

4.4. Video

We compare AugLy’s video augmentations to three existing libraries: moviepy[32], pytorchvideo[6], and

| Library     | # augmentations |
|-------------|-----------------|
| torchvision | 28              |
| AugLy       | 34              |
| Albumentations | 54          |
| imgaug      | 179             |

Figure 7. The image augmentation libraries we chose to compare and their corresponding number of augmentations at the time of writing.

| Library     | # augmentations |
|-------------|-----------------|
| TextAttack  | 13              |
| AugLy       | 16              |
| nlpaug      | 16              |
| textflint   | 55              |

Figure 9. The text augmentation libraries we chose to compare and their corresponding number of augmentations at the time of writing.

| Augmentation     | (1) | (2) | (3) | (4) |
|------------------|-----|-----|-----|-----|
| PerspectiveTransform | 0.333 | 0.032 | 0.076 | **0.013** |
| Sharpen          | 0.159 | 0.021 | 0.141 | **0.005** |
| ColorJitter      | 0.108 | 0.038 | 0.107 | **0.015** |
| Blur             | 0.097 | 0.013 | 0.143 | **0.005** |
| Saturation       | 0.091 | 1.301 | 0.057 | **0.015** |
| Pixelization     | 0.081 | **0.034** | | |
| Brightness       | 0.078 | 0.056 | 0.005 | **0.015** |
| Resize           | 0.056 | 0.014 | 0.050 | **0.006** |
| EncodingQuality  | 0.041 | 0.050 | 0.005 | **0.002** |
| Contrast         | 0.031 | **0.007** | 0.074 | |
| Rotate           | 0.024 | 0.019 | **0.011** | 0.028 |
| Pad              | 0.010 | 0.018 | **0.005** | 0.008 |
| ApplyLambda      | 0.008 | | | **2e-5** |
| Grayscale        | 0.005 | 0.030 | 0.002 | **0.001** |
| HFip             | 0.005 | 0.002 | 0.003 | **0.001** |
| VFip             | 0.003 | 0.001 | 0.002 | **0.001** |
| Crop             | 0.001 | 0.008 | 6e-4 | **2e-5** |

Figure 8. The runtime (in seconds) of image augmentations in (1) AugLy, (2) imaug, (3) torchvision, & (4) Albumentations. Augmentations consistently outperform any other library, likely due to the fact that it uses NumPy arrays as opposed to PIL. We continue to use PIL because it allows for (a) an easy integration with torchvision’s Compose() and (b) better code readability.

| Augmentation     | (1) | (2) | (3) | (4) |
|------------------|-----|-----|-----|-----|
| SimulateTypos    | 0.276 | 0.101 | 0.006 | **4e-4** |
| SwapGenderedWords| 0.102 | | | **0.003** |
| ReplaceSimilarChars | 0.102 | 0.101 | 0.006 | **0.001** |
| SplitWords       | 0.101 | | | **0.101** |
| Contractions     | 0.001 | | | **1e-4** |
| ChangeCase       | | | | **3e-4** |
| InsertPunctuationChars | | | | **1e-4** |
|                  | | | | 6e-4 |

Figure 10. The runtime (in seconds) of text augmentations in (1) AugLy, (2) nlpaug, (3) TextAttack, & (4) textflint.
 augmentations

| Library    | # augmentations |
|------------|-----------------|
| pytorchvideo | 19              |
| moviepy    | 30              |
| vidaug     | 40              |
| AugLy      | 43              |

Figure 11. The video augmentation libraries we chose to compare and their corresponding number of augmentations at the time of writing.

vidaug[16]. See Figure 11 for a comparison of the four libraries in terms of the number of distinct augmentations provided.

Most existing video augmentations either focus on manipulating the spatial dimension or the temporal dimension, as opposed to both. For instance, many individuals apply spatial image augmentations frame by frame onto videos; pytorchvideo provides one such API to do this using the torchvision transforms. Although spatial augmentations are effective, applying temporal augmentations in tandem has been shown to improve performance[30].

Moviepy is more of a general video processing and editing library, but it provides both spatial and temporal manipulations such as changing the speed of the video, trimming, and spatial cropping. vidaug provides similar spatial and temporal augmentations. However, none of these existing libraries provide the option to augment the audio or to perform overlay augmentations which AugLy does provide. AugLy provides a wide array of spatiotemporal augmentations which are common online such as temporally splicing one video into another, simulating a screenshot reshare, and overlaying one video onto another. AugLy is also unique in its multimodal integration, meaning a video’s audio can be transformed then recombined with the video in conjunction with other augmentations.

We benchmark each AugLy video augmentation, as well as the analogues that exist in the other libraries. See Figure 12 for the runtime in seconds of each augmentation in (1) AugLy, (2) moviepy, (3) pytorchvideo, & (4) vidaug. Although other libraries are faster, we continue to use the FFMPEG CLI because we want to be able to process large videos effectively and conserve memory, instead of storing and passing videos in memory as (3) and (4) do.

5. Robustness Evaluation

To demonstrate how AugLy can be used to evaluate robustness, we evaluated a few ImageNet models throughout the years on AugLy augmentations. We were interested to see how robustness has evolved as models’ accuracy has improved, as well as understanding which augmentations the models were particularly vulnerable to. We chose three models to evaluate: VGG16[25], Resnet152[9], and EfficientNet-L2 (Noisy Student)[28].

We evaluated the aforementioned models on the ImageNet validation set, which is commonly used since the test set is not available for download. However, to avoid any potential bias due to overfitting, we evaluated on an additional dataset, ImageNet V2. “ImageNet V2”[22] was put together by researchers with the intention to be a held-out test set for ImageNet that can be evaluated on with no risk of overfitting.

We evaluated the robustness of each model across many different AugLy image augmentations by sampling 250 images from each dataset, computing the top-5 accuracy on those images, and computing the top-5 accuracy when the images are augmented using each augmentation. The change in top-5 accuracy from the baseline (i.e. when the images are not augmented) to the augmented images gives us a measure of how vulnerable the model is to that augmentation. We chose a diverse set of augmentations and set the parameters such that the augmentations were very noticeable but the content of the image was still recognizable to the human eye. See examples of some of the augmentations in Figure 14. The notebook used to perform this robustness evaluation can be found in the AugLy repo at https://github.com/facebookresearch/AugLy/blob/main/examples/imagenet/pwc_imagenet_v1_vs_v2_metrics.ipynb.

In Figure 13, VGG and ResNet are pretty vulnerable to AugLy augmentations across the board. EfficientNet, on the other hand, is much more robust to most
augmentations except for blur and random_noise which cause a larger drop in accuracy. This makes sense considering the augmentations each model was trained on: VGG was trained on augmentations equivalent to AugLy’s crop, hflip, & color_jitter; ResNet was trained on crop, hflip, scale, & color changes similar to color_jitter. EfficientNet was trained using AutoAugment[2], which includes a much wider range of augmentations such as shear_x/y, translate_x/y, rotate, contrast, invert, solarize, posterize, color, brightness, sharpness, and cutout.

Whereas VGG & ResNet were trained on a very limited set of spatial and color-based augmentations, EfficientNet was trained on a larger number of both spatial and color-based augmentations, as well as cutout which is similar to the overlay augmentations in AugLy (but instead of overlaying content over the image, black rectangles are overlaid). However, none of the three models were trained on pixel-level augmentations such as blur, random_noise, or pixelization, which likely explains why all three models are vulnerable to those augmentations. Figure 14 illustrates a few examples from AugLy of the four categories: spatial, color, overlay, and pixel-level augmentations.

We validated that these results are comparable on the ImageNet V2 dataset, shown in Figure 15. Similar to evaluation on the ImageNet validation dataset, VGG and ResNet are quite vulnerable to all augmentations at varying degrees, and EfficientNet is significantly less so with the exception of blur & random_noise.

Figure 16 shows the drop in accuracy on EfficientNet for each augmentation with respect to the original ImageNet validation set and ImageNet V2. The drop in accuracy is close on both datasets for all augmentations, so there is no indication of overfitting on the validation set.

| Augmentation | Original image | Spatial | Color | Overlay | Pixel-level |
|--------------|----------------|---------|-------|---------|-------------|
|              | ![Original image](image1.png) | ![Spatial](image2.png) | ![Color](image3.png) | ![Overlay](image4.png) | ![Pixel-level](image5.png) |
| rotate       | ![Result](image6.png) | | | | blur |
| perspective_transform | ![Result](image7.png) | | | | random_noise |
| color_jitter | ![Result](image8.png) | | | overlay_emoji | |
| brightness   | ![Result](image9.png) | | | overlay_stripes | |

Figure 14. Examples of each different category of image augmentation, as shown on an image from the ImageNet validation set of class 259 (Pomeranian).

6. Conclusion

We presented AugLy, a new multimodal augmentation library with a focus on robustness. We compared each sublibrary (audio, image, text, and video) to other similar aug-
Figure 15. The change in top-5 accuracy caused by each augmentation on each model, computed on a sample of 250 images from the ImageNet V2 “matched frequency” dataset.

Figure 16. The change in top-5 accuracy caused by each augmentation on the EfficientNet-L2 (Noisy Student) model, computed on both the ImageNet validation set & the ImageNet V2 set.

We benchmarked augmentation libraries, assessing the amount of augmentations offered, the kinds of augmentations available, and benchmarking analogous functions to observe their performance. While other libraries may be more performant time-wise, AugLy provides a wide range of unique augmentation that replicate real modifications seen online. Additionally, we evaluated our augmentations on three state-of-the-art image classification models over time, showing that retraining on augmented data is an effective method for building defenses against various attack types.

Acknowledgements

We would like to thank A.K.M Adib, Erik Chou, Aditya Prasad, and Guillermo Sanchez for their contributions to this paper by benchmarking and improving the efficiency of AugLy’s augmentations!

References

[1] Alexander Buslaev, Alex Parinov, Eugene Khvedchenya, Vladimir I. Iglovikov, and Alexandr A. Kalinin. Albumentations: fast and flexible image augmentations, 2020. Information. 2, 3
[2] Ekin D. Cubuk, Barret Zoph, Dandelion Mane, Vijay Vasudevan, and Quoc V. Le. Autoaugment: Learning augmentation strategies from data, 2019. CVPR. 2, 6
[3] Emily Dinan, Angela Fan, Adina Williams, Jack Urbanek, Douwe Kiela, and Jason Weston. Queens are powerful too: Mitigating gender bias in dialogue generation, 2020. EMNLP. 2
[4] Brian Dolhansky, Joanna Bitton, Ben Pfau, Jikuo Lu, Russ Howes, Menglin Wang, and Cristian Canton Ferrer. The deepfake detection challenge (dfdc) dataset, 2020. 3
[5] Matthijs Douze, Giorgos Tolias, Ed Pizzi, Zoe Papakipos, Lowik Chanussot, Filip Radenovic, Tomas Jenecek, Maxim Maximov, Laura Leal-Taixé, Ismail Elezi, Ondřej Chum, and Cristian Canton Ferrer. The 2021 image similarity challenge and dataset, 2021. 3
[6] Haoqi Fan, Tullie Murrell, Heng Wang, Kalyan Vasudev Alwala, Yanghao Li, Yilei Li, Bo Xiong, Nikhila Ravi, Meng Li, Haichuan Yang, Jitendra Malik, Ross Girshick, Matt Feiszli, Aaron Adcock, Wan-Yen Lo, and Christoph Feichtenhofer. PyTorchVideo: A deep learning library for video understanding. In Proceedings of the 29th ACM International Conference on Multimedia, 2021. https://pytorchvideo.org/. 2, 4
[7] Steven Y. Feng, Varun Gangal, Jason Wei, Sarath Chandar, Soroush Vosoughi, Teruko Mitamura, and Eduard Hovy. A survey of data augmentation approaches for nlp, 2021. 1
[8] Tao Gui, Xiao Wang, Qi Zhang, Qin Liu, Yicheng Zou, Xin Zhou, Rui Zheng, Chong Zhang, Jiacheng Ye Qinzhuo Wu, Zexiong Pang, Yongxin Zhang, Zhengyan Li, Ruotian Ma, Zichu Fei, Ruijian Cai, Xingshu Hu Jun Zhao, Zhiheng Yan, Yiding Tan, Yuan Hu, Qiyuan Bian, Zhihua Liu, Bolin Zhu, Shan Qin, Xiaoyu Xing, Jinlan Fu, Yue Zhang, Minlong Peng, Xiaoqing Zhong, Zhongyu Wei Yaqian Zhou, Xipeng Qiu, and Xuanjing Huang. Textflint: Unified multilingual robustness evaluation toolkit for natural language processing, 2021. arXiv preprint arXiv:2103.11441. 2, 4
[9] Kaiming He, Xiangyu Zhang, Shaoqing Ren, and Jian Sun. Deep residual learning for image recognition, 2015. 5
[10] Dan Hendrycks, Steven Basart, Norman Mu, Saurav Kadamth, Frank Wang, Evan Dorundo, Rahul Desai, Tyler Zhu, Samyak Parajuli, Mike Guo, Dawn Song, Jacob Steinhardt, and Justin Gilmer. The many faces of robustness: A critical analysis of out-of-distribution generalization, 2021. ICCV. 1
[11] Dan Hendrycks and Thomas Dietterich. Benchmarking neural network robustness to common corruptions and perturbations, 2019. ICLR. 1
[12] Dan Hendrycks, Norman Mu, Ekin D. Cubuk, Barret Zoph, Justin Gilmer, and Balaji Lakshminarayanan. Augmix: A simple data processing method to improve robustness and uncertainty, 2020. ICLR. 2
[13] Iver Jordal. Audiomentations, Aug. 2021. 3
Appendix

| Augmentation          | (1) | (2) | (3) | (4) |
|-----------------------|-----|-----|-----|-----|
| Harmonic              | 2.897 |     |     |     |
| Percussive            | 2.897 |     |     |     |
| PitchShift            | 1.238 | 0.372 | 0.651 |     |
| TimeStretch           | 0.415 | 0.053 | 0.121 |     |
| Reverb                | 0.271 |     |     |     |
| Tempo                 | 0.195 |     |     |     |
| AddBackgroundNoise    | 0.048 | 0.019 |     |     |
| PeakingEqualizer      | 0.048 |     |     |     |
| InsertInBackground    | 0.044 |     |     |     |
| ChangeVolume          | 0.035 | 3e-5  | 0.034 | 0.004 |
| HighPassFilter        | 0.017 | 3e-4  | 0.017 | 0.413 |
| ToMono                | 0.016 |     | 0.022 |     |
| Normalize             | 0.015 | 4e-5  | 0.043 | 0.004 |
| LowPassFilter         | 0.014 | 5e-4  | 0.013 | 0.163 |
| Clicks                | 0.009 |     |     |     |
| Loop                  | 0.006 |     |     |     |
| InvertChannels        | 0.003 |     |     |     |
| ApplyLambda           | 0.003 |     |     |     |
| Speed                 | 0.002 | 6e-5  |     | 0.003 |
| Clip                  | 0.002 |     |     |     |

Figure 17. The runtime (in seconds) of audio augmentations in (1) AugLy, (2) pydub, (3) torchaudio, & (4) audiomentations.
| Augmentation                  | (1)   | (2)   | (3)   | (4)   |
|------------------------------|-------|-------|-------|-------|
| ShufflePixels                | 1.600 |       |       |       |
| PerspectiveTransform         | 0.333 | 0.032 | 0.076 | 0.013 |
| Sharpen                      | 0.159 | 0.021 | 0.141 | 0.005 |
| ApplyPILFilter               | 0.117 |       |       |       |
| ColorJitter                  | 0.108 | 0.038 | 0.107 | 0.015 |
| Blur                         | 0.097 | 0.013 | 0.143 | 0.005 |
| Saturation                   | 0.091 | 1.301 | 0.057 | 0.015 |
| ChangeAspectRatio            | 0.091 |       |       |       |
| Skew                         | 0.084 |       |       |       |
| OverlayStripes               | 0.083 |       |       |       |
| Pixelization                 | 0.081 | 0.034 |       |       |
| Brightness                   | 0.078 |       | 0.056 | 0.005 |
| OverlayOnto Screenshot       | 0.064 |       |       |       |
| Scale                        | 0.059 |       |       |       |
| Resize                       | 0.056 | 0.014 | 0.050 | 0.006 |
| OverlayOnto BackgroundImage  | 0.049 |       |       |       |
| EncodingQuality              | 0.041 | 0.050 | 0.002 |       |
| ConvertColor                 | 0.039 |       |       |       |
| MaskedComposite              | 0.038 |       |       |       |
| Contrast                     | 0.031 | 0.007 | 0.074 |       |
| Opacity                      | 0.029 |       |       |       |
| OverlayText                  | 0.027 |       |       |       |
| MemeFormat                   | 0.025 |       |       |       |
| Rotate                       | 0.024 | 0.019 | 0.031 | 0.028 |
| OverlayImage                 | 0.023 |       |       |       |
| Pad                          | 0.010 | 0.018 | 0.005 | 0.008 |
| ApplyLambda                  | 0.008 |       |       | 2e-5  |
| PadSquare                    | 0.008 |       |       |       |
| OverlayEmoji                 | 0.006 |       |       |       |
| Grayscale                    | 0.005 | 0.030 | 0.002 | 0.001 |
| HFlip                        | 0.005 | 0.002 | 0.003 | 0.001 |
| VFlip                        | 0.003 | 0.001 | 0.002 | 0.001 |
| ClipImageSize                | 0.002 | 0.008 | 0.008 | 2e-5  |
| Crop                         | 0.001 | 0.008 | 0.008 | 2e-5  |

Figure 18. The runtime (in seconds) of image augmentations in (1) AugLy, (2) imgaug, (3) torchvision, & (4) Albumentations.

| Augmentation                  | (1)   | (2)   | (3)   | (4)   |
|------------------------------|-------|-------|-------|-------|
| SimulateTypos                | 0.276 | 0.101 | 0.006 | 4e-4  |
| SwapGendered Words           | 0.102 |       |       |       |
| Replace FunFonts             | 0.102 |       |       |       |
| Replace Similar UnicodeChars | 0.102 |       |       |       |
| Replace UpsideDown           | 0.102 |       |       |       |
| MergeWords                   | 0.102 |       |       |       |
| Replace SimilarChars         | 0.102 | 0.101 | 0.006 | 0.001 |
| SplitWords                   | 0.101 | 0.101 |       |       |
| ReplaceWords                 | 0.101 |       |       |       |
| GetBaseline                  | 0.101 |       |       |       |
| Contractions                 | 0.001 | 1e-4  | 2e-4  |       |
| ChangeCase                   | 4e-4  | 3e-4  |       |       |
| Insert Punctuation Chars     | 1e-4  | 0.002 | 6e-4  |       |
| Insert Whitespace Chars      | 7e-5  |       |       |       |
| Replace Bidirectional        | 7e-5  |       |       |       |
| InsertZero WidthChars        | 6e-5  |       |       |       |
| ApplyLambda                  | 6e-5  |       |       |       |

Figure 19. The runtime (in seconds) of text augmentations in (1) AugLy, (2) nlpaug, (3) TextAttack, & (4) textflint.
| Augmentation                  | (1)   | (2)   | (3)   | (4)   |
|------------------------------|-------|-------|-------|-------|
| ReplaceWith ColorFrames      | 2.241 |       |       |       |
| Loop                         | 2.015 | 2e-4  |       |       |
| Perspective Transform AndShake| 2.015 |       |       |       |
| BlendVideos                  | 1.905 |       |       |       |
| ReplaceWith Background       | 1.746 |       |       |       |
| OverlayText                  | 1.677 |       |       |       |
| OverlayShapes                | 1.659 |       |       |       |
| TimeDecimate                 | 1.636 |       |       |       |
| OverlayOnto Screenshot       | 1.632 |       |       |       |
| OverlayDots                  | 1.631 |       |       |       |
| MemeFormat                   | 1.610 |       |       |       |
| InsertIn Background          | 1.605 |       |       |       |
| OverlayOnto BackgroundVideo  | 0.781 |       |       | 0.016 |
| Shift                        | 0.773 |       | 1.996 |       |
| Pixelization                 | 0.662 |       |       |       |
| AugmentAudio                 | 0.625 | 0.001 |       |       |
| OverlayEmoji                 | 0.501 |       |       |       |
| Concat                        | 0.467 |       |       |       |
| AudioSwap                     | 0.445 |       |       |       |
| VStack                        | 0.435 |       |       |       |
| Overlay                      | 0.406 |       |       |       |
| HStack                       | 0.400 |       |       |       |
| Pad                           | 0.400 | 0.018 |       |       |
| TimeCrop                     | 0.395 |       | 1e-5  |       |
| Trim                          | 0.386 |       |       |       |
| ChangeAspectRatio            | 0.368 |       |       |       |
| Crop                         | 0.352 | 9e-5  | 2e-5  |       |
| Rotate                       | 0.336 | 1e-4  | 0.202 | 0.275 |
| Blur                         | 0.307 | 0.140 | 0.179 |       |
| VFlip                        | 0.297 | 9e-5  | 0.151 | 2e-5  |
| AddNoise                     | 0.297 |       |       | 0.036 |
| Resize                       | 0.289 | 0.015 |       |       |
| Scale                        | 0.284 |       |       |       |
| FPS                          | 0.271 |       |       |       |
| ChangeVideo Speed            | 0.269 | 1e-4  | 1e-4  |       |
| HFlip                        | 0.269 | 1e-4  | 0.152 | 2e-5  |
| Grayscale                    | 0.266 | 0.047 | 0.081 |       |
| Contrast                     | 0.264 |       |       |       |
| Encoding Quality             | 0.262 |       |       |       |
| ColorJitter                  | 0.262 | 0.035 | 0.077 |       |
| Brightness                   | 0.258 | 0.050 |       |       |
| RemoveAudio                  | 0.255 |       |       |       |
| ApplyLambda                  | 5e-4  |       |       |       |

Figure 20. The runtime (in seconds) of video augmentations in (1) AugLy, (2) moviepy, (3) pytorchvideo, & (4) vdaug.