California Institute of the Arts

label.art - a data labeling platform combining human intelligence with AI and automation to annotate all varieties of data

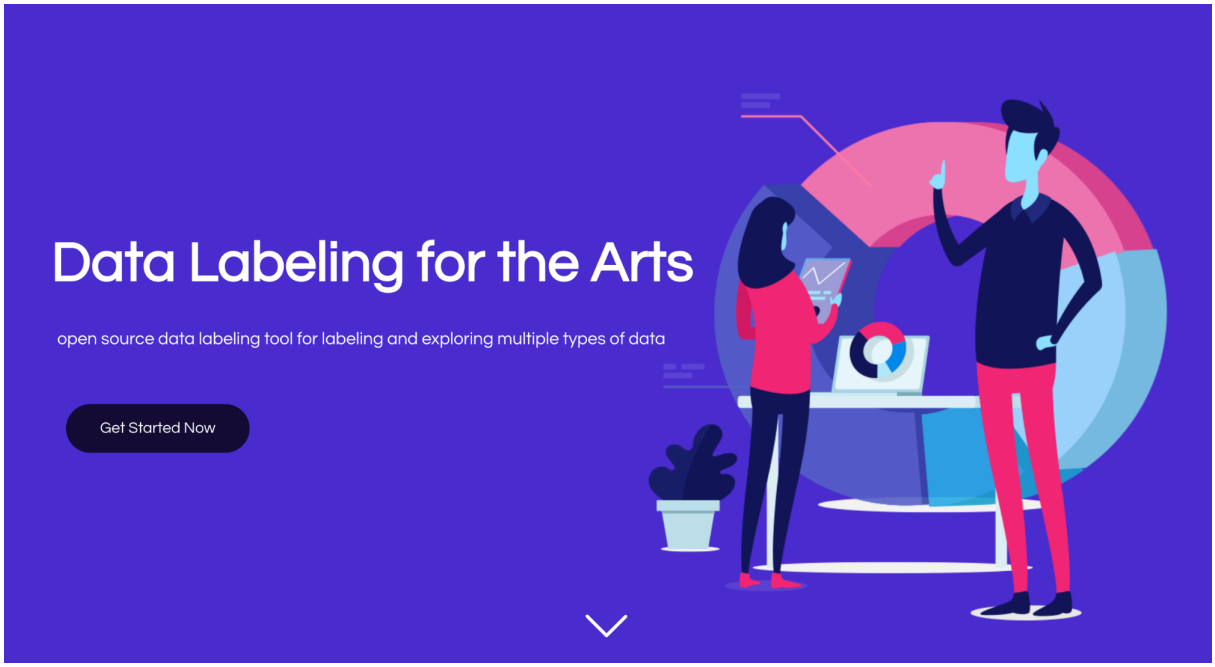
by Max Fishman

A thesis submitted in fulfillment for the degree of Bachelors of Fine Arts Herb Alpert School of Music, Music Technology: Interaction, Intelligence & Design 2022.

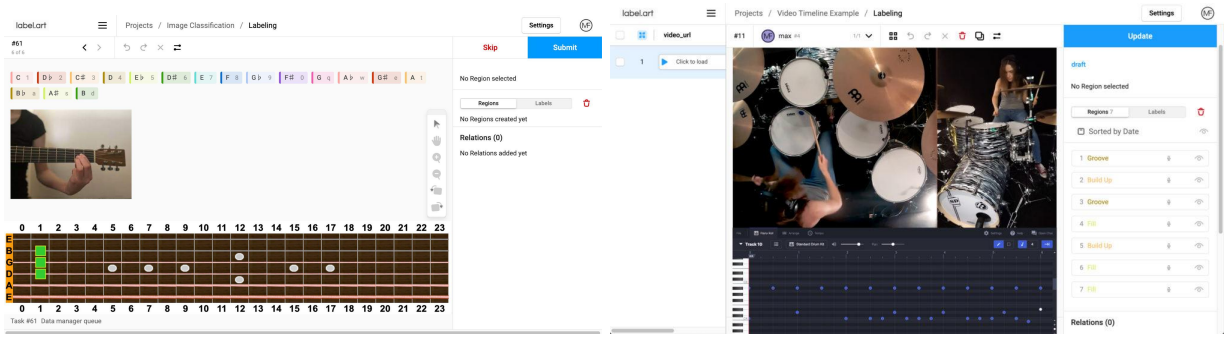
Abstract

This thesis paper addresses extensive research on how to augment the traditional data labeling workflow and annotation processing for machine learning and computer vision research, specifically for the development of new AI/ML tools for audio and visual artists.

This paper provides background and history of image classification, applications of supervised learning, and issues with current data labeling approaches for modern artistic applications. There will be an emphasis on the data collection process, the importance of accurate and unbiased data labeling, and the use of open-source software and cloud computing infrastructures for training unique neural networks at scale for knowledge specific tasks.



Chapter 1 - Introduction



[Piano Roll and Guitar Neck music GUI interfaces for data labeling and entry]

The market for AI and machine learning relevant data preparation solutions was over $1.5B in 2019, growing to $3.5B by the end of 2024[1]. Most tools currently available to the public are for generalized machine learning tasks, such as self-driving, object and image detection, and voice-to-text. An increasing number of training requirements are becoming more knowledge-specific as machine learning tasks are becoming more specialized. Therefore, we need a unified platform to develop the labeling and data preparation interfaces and workflows that will power the future of machine learning and artificial intelligence for artists.

Label.Art aims to provide artists and researchers with the tools they need to accurately and efficiently label vast amounts of data for machine learning and artificial intelligence computer tasks. This paper outlines the reasons why such software needs to be accessible and free, the ethical and moral implications of building AI/ML software, and the technical aspects to deploy the software for one's personal use.

Chapter 2 - Early Internet / Open Source / Community Labeling

In developing this new data labeling platform, I harnessed decades of research from the open source software community. I knew I would not need to "reinvent the wheel", and would just need to adapt its use case. "Open source" is used to describe a class of software that is subject to a license agreement that grants special rights and obligations to the Licensee.

Software as intellectual property is protected by copyright law. Traditional commercial software developers use the copyright laws to sell or provide their software under license to the buyer. These licenses often prevent the user of the software from inspecting the original source code, modifying the software, copying or sharing. This type of software is often referred to as "closed source" or "proprietary" software.

Open Source software is also protected by copyright. However, an Open Source or FOSS (Free Open Source Software) license grants irrevocable rights to the software user to receive and inspect the original source code, to modify the source code in any way, and to freely distribute the original or derivative works of the software without compensating the copyright holder. One of FOSS' most important and radical requirements, is when the user of the software, and its derivatives, sell or give the software to someone else, they must pass along all of the same provisions and rights of the original FOSS license.

In the context of FOSS, *free* does not mean without remuneration, or at no cost. It is intended to signify *freedom,* as in the freedoms to inspect, modify, distribute, etc. Since FOSS essentially protects user rights, instead of creator rights, it is commonly referred to as a *copyleft* license. The concept and

term was invented by Richard Stallman in 1983, when he formed the GNU project, and later the Free Software Foundation. (Fogel, 2016)

Open Source software has several powerful advantages over proprietary software. Groups of programmers can contribute to the software and each benefits from the contributions of the group. They are also protected from subsequent contributors turning their effort into proprietary software. Arguably the most well known Open Source project is the Linux operating system.

I used several open-source and open-standard software for this project. I used the **LAMP** stack - **L**inux, **A**pache, **M**ySQL and **P**HP, which consist of an operating system, web server, database, and the programming language PHP. I integrated three other open source projects, Wordpress, Docker and Label Studio, as well as several open standards.

Table 1 (below) contains a list of Open Source software used in my project.

Open Source Project	Function	Open Source License
Linux	Operating System	GNU General Public License (GPL)
Apache	Web Server	Apache License 2.0
mySQL	Database	GNU General Public License (GPL)
PHP	Programming Language	The PHP License, version 3.01
Wordpress	Content Management System	GPLv2
Label Studio	Data Annotation and ML Packages	Apache License 2.0
Python	Programming Language	Python 3.4.0 license
ReactJS	UI JavaScript Library	MIT License
Docker	OS virtualization	Open Sourced in 2013

The use of Open Source software allowed me to build a working data labeling platform leveraging the millions of lines of software written by the Open Source community.

Machine Learning approaches in an artists' workflow

Training machine learning models usually involves a mixture of supervised, unsupervised, and semi-supervised learning. Each of these different processes have human labelers involved at different levels. Here are some examples of common training techniques, and how they could fit into an artist's workflow using Label.Art to first label and annotate their data.

The machine learning backend included in Label.Art uses open source projects including uWSGI, Supervisord, and Redis Queue to handle machine learning tasks in python. Using the included Machine Learning SDK, you can set up any model as the backend to send and receive predictions as labeling happens in real time. This allows users to integrate into common machine learning tools like Pytorch, GPT, Tensorflow, etc. If run locally, a user can easily deploy the provided example ml backends using docker containers communicating with each other via the machines localhost connection and ports.

Supervised learning is a type of machine learning which trains algorithmic models on known input and output data (i.e., clean and labeled data samples). Supervised learning can classify items based on a ground truth supplied to the original model. This approach is beneficial in classification problems like identifying a speaker in a group conversation, or for doing sonic classification of musical instruments in an audio file. Using label.arts' Speaker Segmentation template, human labelers can supervise their audio waveform in real-time, and label regions of audio and create relationships and hierarchy between data points.

Semi-supervised learning is a combination of supervised and unsupervised learning. Semi-supervised models use small amounts of clean and labeled training data, along with large quantities of unlabeled training data, to overcome the disadvantages of both supervised and unsupervised approaches. This type of training uses high-quality transcriptions and annotations, on a small portion of the data, to provide a strong reference model that is used for the rest of the data. However, the effectiveness of this method depends on the quality of the model created on the initial labeled data.

Using label.art you can import any pre-annotated data into your network. With these pre-labeled data points, Label.Art can assist human labelers by providing a predicted annotation for common tasks like image segmentation and object recognition, using bounding boxes, polygons, key points, and ellipses. By making a POST request to the predictions endpoint of the Label Studio API, you can prompt the ML_backend to create predictions to display on the frontend for the labeling team. This process can also be automated using a Webhook payload to trigger a new training cycle or update annotations across labeling team members in the background.

Unsupervised learning is the process of teaching a machine learning model to use data, that has neither been classified nor labeled, and then letting the program use that data without supervision. The goal of unsupervised learning is to have the machine group the unsorted information according to similarities, patterns, and differences, without any prior training of the data.

Unsupervised learning helps solve clustering and associative problems, especially in audio recommendation engines where similar songs can be grouped based on BPM, pitch, and instrumentation. Using the tools built into label.art, you can clean and prepare clean labeled data to validate and train your unsupervised model further, or use it to fix labeling mistakes made by an unsupervised model.

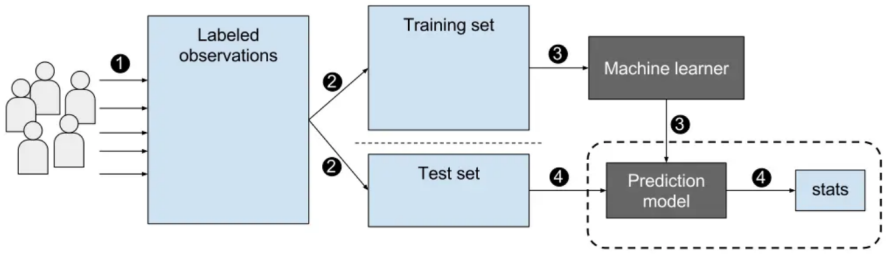


Fig. 1 - Example of Supervised Machine Learning (Nvidia)

Chapter 3 - Accurate and Unbiased Data Labeling

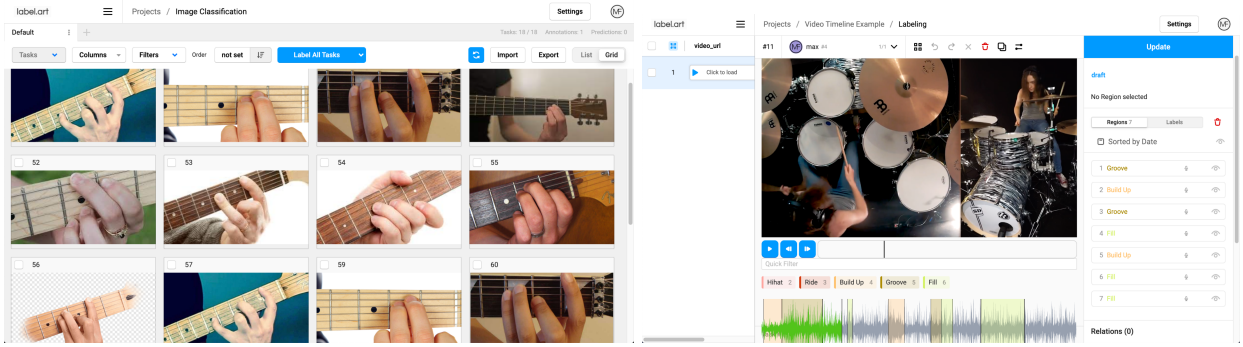


Figure 1 - (left) Image Classification of Guitar Chords (Right) Video + Audio Timeline labeling interface

When building any system, especially one that is open source and representative of a community, one must ensure that the data and problems focused on are all-encompassing and inclusive, in order to ensure an accurate and unbiased data set.

Hypothetically, assume we are teaching a computer vision system to identify chords on a guitar neck using images. If the data set is made up of primarily light skinned hands and fingers, we will inevitably run into problems trying to train our network to be a generalized solution. It is also important to

consider the overall size of the dataset, because if the data set is too small, or repetitive, one may experience “mode collapse”.

In the 2018 project “Gender Shades”, an intersectional approach was applied to appraise three gender classification algorithms, including those developed by IBM and Microsoft. Subjects were grouped into four categories: darker-skinned females, darker-skinned males, lighter-skinned females, and lighter-skinned males. All three algorithms performed the worst on darker-skinned females, with error rates up to 34% higher than for lighter-skinned males (Figure 2).

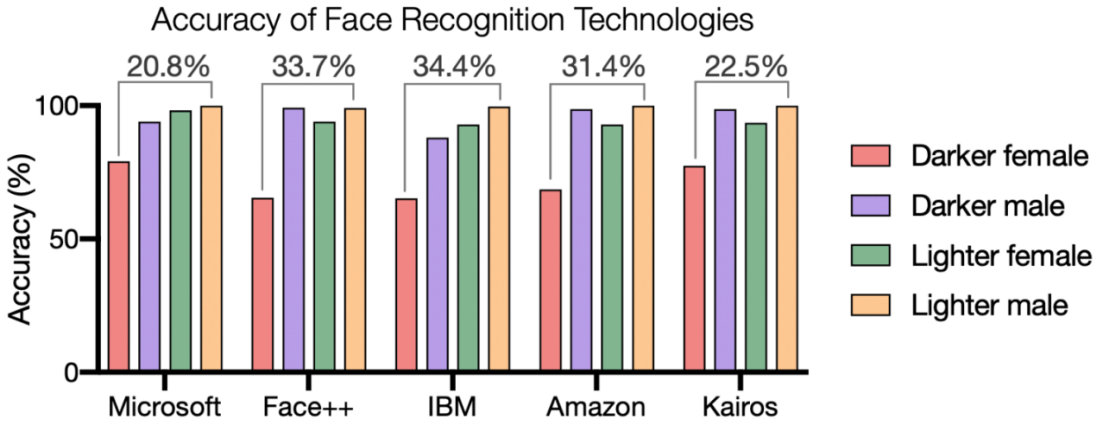


Figure 2

Independent assessment by the National Institute of Standards and Technology (NIST) has confirmed these studies, finding that face recognition technologies across 189 algorithms are least accurate on women of color.

Identifying and remediating sources of AI bias is critical given the types of applications now being deployed. One particular concern is the rapid adoption of artificial intelligence and ML technologies in law enforcement. Facial Recognition Technology, or FRT programs, used by law enforcement in identifying crime suspects, are substantially more error-prone on facial images depicting

darker skin tones, and females, as compared to facial images depicting Caucasian males. This bias has resulted in a higher percentage of female and dark skinned people being wrongfully investigated by police. The Equal Protection Clause of the 14th Amendment protects citizens from discrimination under the law, or through government action. However, the federal Government Accountability Office reported in July 2021, that 42 federal agencies that employ law enforcement officers have used facial recognition technology in one form or another. Given that this technology in use today has built in bias, use of it is inherently unconstitutional.

While most studies confirm the prevalence of AI / ML or algorithmic bias, the public perception of FRT is very different. Most people believe that computers are immune from human bias, and computers will 'calculate' the correct answer. For example, following a theft of a watch at a retail store, the police will review the security camera recording and capture a screen image of the suspect removing the watch and leaving the store. Using an FRT service, they compare the security camera image using one of the commercial FRT services. Based on the MIT study results, if the suspect was a white male, the chance of misidentifying the suspect would be 0.8% , while in the case of a dark skin female, the error rate would be 34.7%. From a racial bias perspective, innocent dark skinned females are 43 times more likely to be detained than an innocent white male.

"On the Internet, and in our everyday uses of technology, discrimination is also embedded in computer code, and increasingly in artificial intelligence technologies that we are reliant on, by choice or not. "

- Safiya Umoja Noble

Fig 1 - Example of Mode Collapse in an image GAN

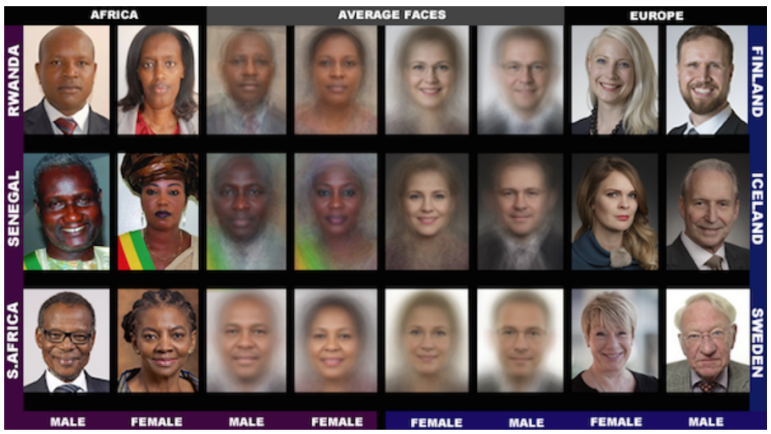


Fig 2 -Example of "Unbalanced" Data Set

Chapter 4 - The Labeling Team

In the future, what will the training of artificial neural networks look like? How will humans be involved in the process? How much can we automate reliably? What considerations should be given to ensure accurate, consistent and unbiased data out of your network when working with labeling arts specific data sets?

Data preparation and labeling tasks represent over 80% of the time consumed in most AI and Machine Learning projects. The access to open-source pre-trained models moves an increasing amount of the human labeling task to automation over time or shifts the human labelers focus to more complex tasks. (https://www.cognilytica.com/document/data-preparation-labeling-for-ai-2020/)

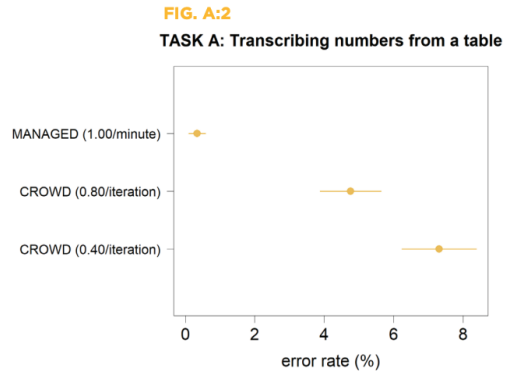
For example, Tesla Motors recently laid off about 200 employees working as in-house data labelers. Their work is critical for Tesla to achieve its promised “full self-driving” capability. Tesla employs thousands of data labelers across the country all working to train specific neural networks designed to tackle aspects of the self-driving task.

In recent years, the Tesla team has moved to a newer “auto-labeling” and “vector-space” or“simulated” approaches. Using this combination of approaches, instead of needing to label eight single image sources (cameras), the human labeler can annotate in “vector space”, and with one click annotate all eight image sources. This process allows one human labeler to do more with the vast amount of data that the fleet of millions of cars collects, and send it back for closer human review.

This process saves time and reduces cost, and additionally, you can use simulated data to create edge cases, or very specific data needed for training. Tesla and the other autonomous self-driving companies need to accumulate many millions of real world examples that are both clean and diverse to actually train their neural networks effectively. Therefore, the move to 3D or 4D labeling from 2D image labeling was necessary. I believe that the same tools being developed to help Tesla's driving team could also be used to accelerate labeling and training of many machine learning algorithms.

A huge amount of data and labeling is necessary to train reliable and unbiased neural nets. Outsourcing data labeling for generalized tasks (like driving a car) is a viable option that can work and scale. However, when trying to find data labelers to train neural networks to identify something more nuanced, like advanced musical theory concepts, artistic brush stroke techniques, or prepare and clean images for AI artmaking, it becomes harder and more expensive to outsource that very niche and creative knowledge, and have confidence in the data being generated by the team.

A report from CloudFactory and Hivemind found that, in a simple transcription task, “the error rate of labeled data from outsourced services ranges from 7% for a simple transcribing task to 80% for rating sentiment from reviews. The effect is particularly pronounced when the workers are paid less.” The report also shows that “crowdsourced workers had an error rate of more than 10x the managed workforce.”



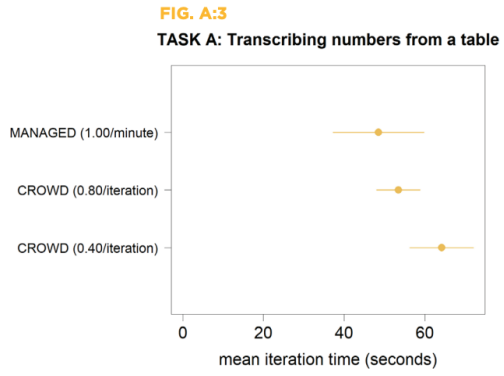


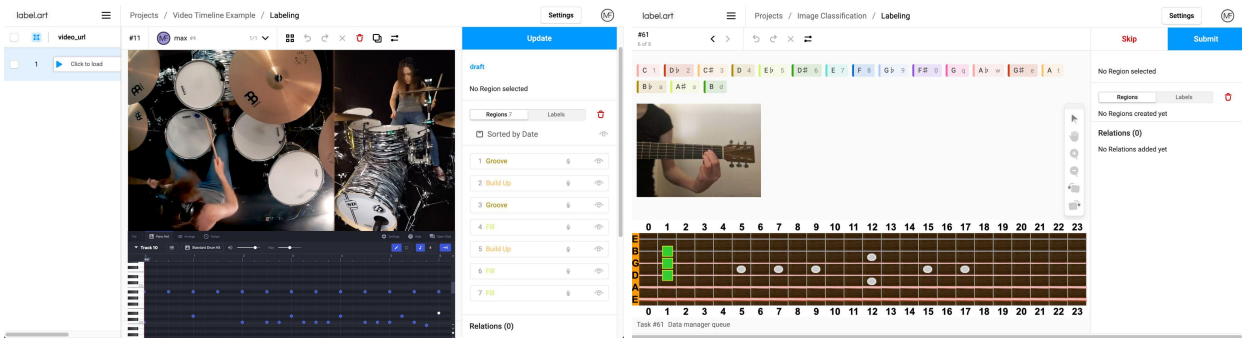
Fig A:2- Error Rate of Simple Transcription Task Fig A:3 - Mean Time of Simple Transcription Task

What can be done to help both established managed teams and crowdsourced communities improve their error rate and mean iteration time when collecting, cleaning, and training knowledge specific data? How can we build the tools so that crowdsourced groups and communities have access to better tools for working collaboratively, efficiently and effectively?

This problem can be resolved by a network of artists and creatives who together become an asset to solve complex machine learning and AI problems. Proper communication with your labeling team, and taking the time to ensure consistency among team members, is crucial in avoiding inconsistencies in the labeling task. These inconsistencies can be hard to detect, and can go unnoticed, if not managed

properly. Using features built into Label.Art, you are able to communicate directly with your data labeling team, and make sure everyone is in the loop.

Chapter 5 - The labeling workflow



Once deployed, you are able to integrate your training pipeline with your data labeling workflow by adding a machine learning (ML) backend to Label.Art. Thereafter, you can set up your machine learning frameworks to pre-label data by letting models predict labels. Then human annotators perform further refinements as needed.

You can also auto-label by letting pre-trained models create automatic annotation predictions. In addition, you are able to do online learning by simultaneously updating your model, while new annotations are created by the labeling team.This feature allows you to retrain your model on-the-fly. You

can perform active learning by selecting example tasks that the model is uncertain how to label for your annotators to manually label.

You can dynamically pre-annotate data based on model inference results, and retrain or fine-tune a model based on recently annotated data. Additionally, you can add as many labelers to your project to work collaboratively, and check their work against pre-trained models. With many annotation templates available, you can get started with labeling quickly, or build a more custom GUI using XML tags and style.

The Label Studio project is available as modular packages that you can plug into your existing workflows. The backend uses Python and Django to perform labeling tasks and is installed via pip package. The frontend is a Javascript web application built using React and MobX. The Data Manager module is a javascript web application built using React for managing labeling tasks. Finally, the machine learning backends (built with Python), predict labels at different parts of the labeling process, and post predictions setup in the frontend UI, via the “Machine Learning” tab for any projects. You can add multiple models for predictions or training. This module is also responsible for running and initializing the port on localhost for ML training.

Chapter 6 - Synthetic Data / Simulation / Auto-Labeling

Fig 1 - Tesla FSD Simulation for Training

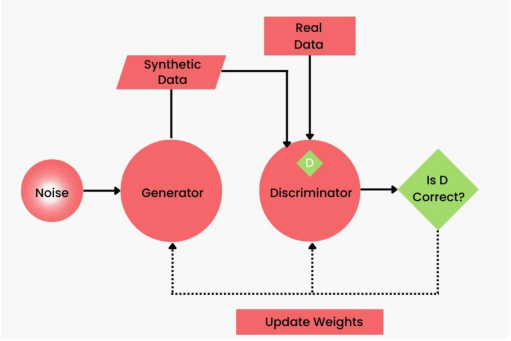


Fig 2 - Example of Using Synthetic Data in a RNN

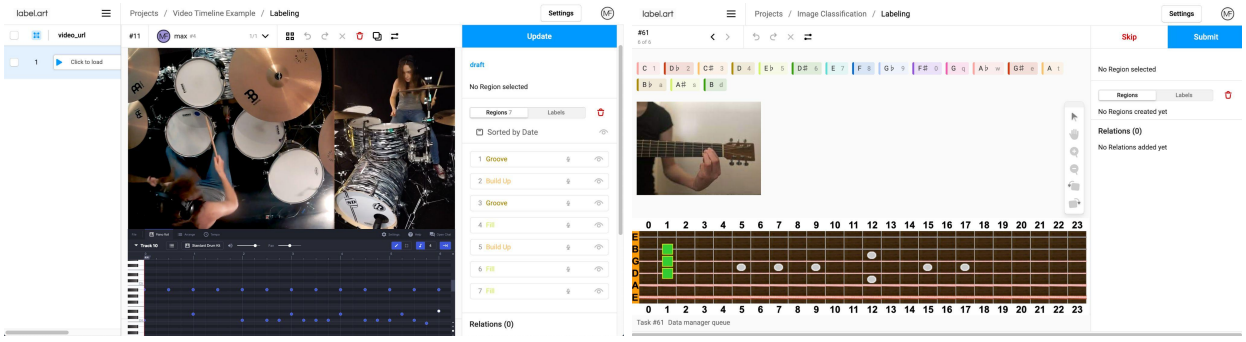
While most of the data being used to train neural networks currently comes from scrapping the internet or from real life data, it is predicted that, “By 2024, 60% of the data used for the development of AI and analytics projects will be synthetically generated”(1). The use of synthetic data has seen exponential growth in recent years, with projects like Microsofts’ SmartNoise or Amazon Go’s cashier-less stores that have been trained using a synthetic dataset. “As our application improved in accuracy — and we have a very highly accurate application today — we had this interesting problem that there were very few negative examples, or errors, which we could use to train our machine learning models,” Dilip Kumar, VP of Amazon Go, said, “So we created synthetic datasets for one of our challenging conditions, which allowed us to be able to boost the diversity of the data that we needed.” (2).

The need for both cleanly labeled real and simulated data is going to become crucial for training effective ml models. If we are teaching a ML model to identify and track the path of a drummers’ sticks, imagine how much easier it is to artificially generate 100,000 images of drummers playing from all angles than to collect those images in the real world, or by scraping and cleaning images found online.

Chapter 7 - Conclusions + Future Project Innovations

This paper's aim was to investigate how to best build systems for the effective labeling of vast amounts of highly specific datasets for ML/AI art projects. The research involved an exploration into machine learning, computer vision, deep-learning, racial and economic impacts of AI, open-source community, and many modern web technologies. It also explored machine learning architecture, data acquisition, virtual environments and containers, and cloud computing.

Future project innovations and work will involve the creation of more robust audio and music tools for annotation. Tools capable of annotating using midiOSC, or traditional music notation GUI, for fast and quick data entry, as well as research into more advanced machine learning backends. Possible software improvements could include more features for error calculation, a front end system to build ML backends, as well as optimization across the various components (Backend, FrontEnd, ML SDK, MMDetection, ML Backend etc).



`

https://blogs.gartner.com/andrew_white/2021/07/24/by-2024-60-of-the-data-used-for-the-development-of-ai-and-analytics-projects-will-be-synthetically-generated/

https://venturebeat.com/ai/amazon-go-uses-synthetic-data-to-train-cashierless-store-algorithms/

Perens, B. (1999). The Open Source Definition in Open Sources: Voices from the Open Source Revolution. 1st edition. Available at http://www.oreilly.com/openbook/opensources/book/perens.html (accessed in June 2018).

https://onlinelibrary.wiley.com/doi/10.1111/jwip.12114

- Avram, Abel (March 27, 2013). "Docker: Automated and Consistent Software Deployments". *InfoQ*. Retrieved August 9, 2013.

Citation. Noble, S. U. (2018). Algorithms of oppression: How search engines reinforce racism. New York University Press.

https://go.cloudfactory.com/hubfs/02-Contents/3-Reports/Crowd-vs-Managed-Team-Hivemind-Study.pdf