MUSIC RECOMMENDATION SYSTEM

TEAM MEMBER

A0215513R ZHANG JIUYUN

A0186896B MA ZEYU

A0215300A JIANG HAO

MASTER OF TECHNOLOGY PROJECT REPORT

Hao Jiang[1,*], Zhang JiuYun[1,*], Ma ZeYu[1,*]

National University of Singapore

Contents

*Corresponding author.

Email addresses: haojiang@u.nus.edu (Hao Jiang), e0535603g@u.nus.edu (Zhang JiuYun), e0321134@u.nus.edu (Ma ZeYu)

1. EXECUTIVE SUMMARY

Automatic video editing is receiving increasingly attention as the digital camera technology develops further and social media sites such as YouTube and TikTok become popular. Background music selection is one of the key elements to make the generated video attractive. In this work, we propose a system for background music recommendation based on semantic analysis and rule engine between video and music.

1.1. market research

Now, in people's daily fragmented time, it is very common to use short video software to share their experiences, beautiful scenery, or watch video shared by other. There are many popular App, such as tik-tok, Lasso or Instagram. In the past, people might prefer to watch TV or listen to music. With the development of mobile technology and the improvement of network connection speed, live broadcast and short video are replacing the previous status of TV and Walkman. For short videos, the main feature is the ability to combine vision and hearing to form relatively strong sensory stimulation in a short time. At the same time, the requirement of production threshold is low, but it still have strong social attributes, which attracting a large number of users. The trend of users in relevant short video app is shown in Figure.1.

According to Figure.1, the users in 2019 is almost ten times compare with 2017. Short video filed has a huge market in future.

1.2. problem in selecting BGM in short video

In the field of short video, ML/DL technology also shows great potential. For example, the recommendation algorithm that tiktok is famous for, or the image processing technology in the video, the optimization of the face and background.

However, for the music recommendation part, AI technology has not been widely used. Imagine that when you make a video and plan to share it, the software will only recommend some music based on your age, gender, preference, or the current hottest music, and it will not be linked to your video content. The result of this is that algorithm output music is often not suitable with your video. If we have a algorithm to make a more suitable music recommendation system for this video, then we believe the produced video must be more attractive.

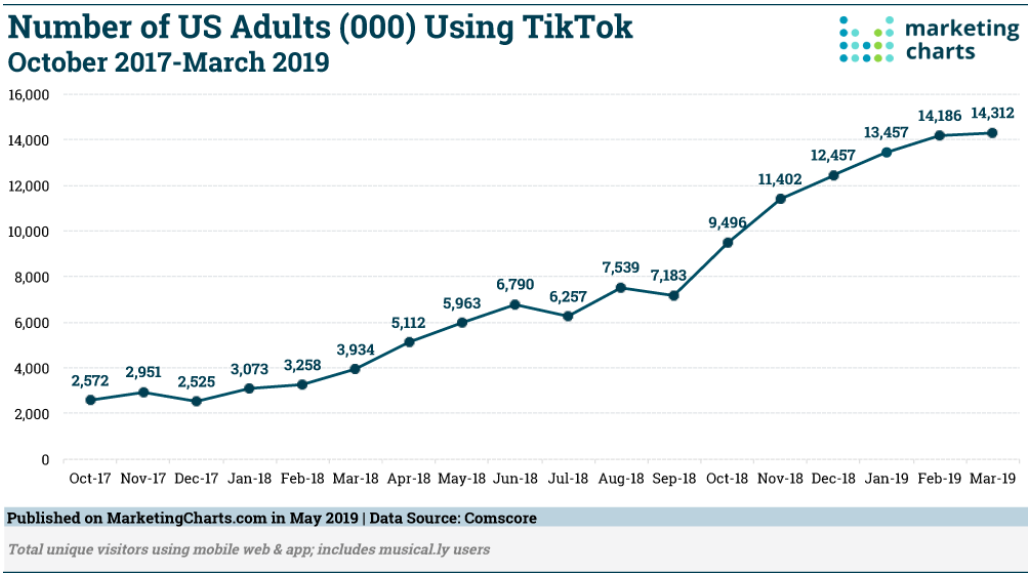


Figure 1: Increasing number of users of TikTok

1.3. product plan

In our project, we hope to solve the problem that the music recommended in short video production does not meet the video content. With our product,when the user inputs a video, our product can match the most suitable BGM for it and output the combined video. In this process, our algorithm will extract some frames from the video, analyze the brightness of each frame and do image classification, then use the rule engine to find the most matching music. We also want to use GA (genetic algorithms) to improve our matching algorithm base on the user viewing time. The pipeline of our recommendation system is shown in Figure.4

In the future, in addition to the content of the video itself, we may combine users' own data, preferences, or popular music of the month and year to make a more reasonable detection.

1.4. business case

After we done our system, we do a survey around our friends and parents, which have 15 people include 5 young, 5 middle-aged and 5 old-aged, thus the test population samples are equally distributed. The survey conclusion is shown in Figure.2. We can know in this figure that around two third of our test samples think this product is useful and can bring help when creating or uploading videos.

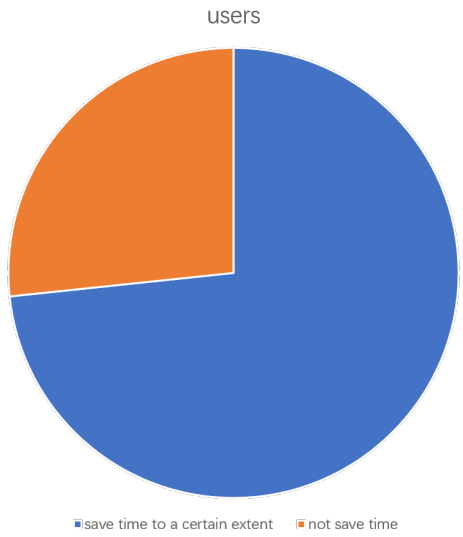


Figure 2: Users feedback of whether the system is useful (save time)

2. Automatically Recommend Music System

This web application is build with two layers: application layer for users to upload their videos and synthesize the recommended music with the video, hybrid reasoning layer for decomposing videos into labels and reasoning the correct music type.

For application layer, we combines python Flask framework with FFmpeg extension to provide a simple, flexible and user-friendly interface for users.

For hybrid reasoning layer, we apply genetic algorithm to improve the structure of reasoning system which is established by expert interviews. Additionally, a multi-label classifier is built to extract labels from video. Our system architecture is shown as follows: Figure.3

The workflow of our automatically recommend music system can be decomposed into four main stages: (1) Multilabel classification algorithm (2) Rule Engine (3) GA (4) UI The flow graph of our system pipeline is shown in Figure.4,

2.1. Multilabel classification

Our multilabel classification model is set by 'Multi-Label Image Recognition with Graph Convolutional Networks' [1], further more, we use opencv library for

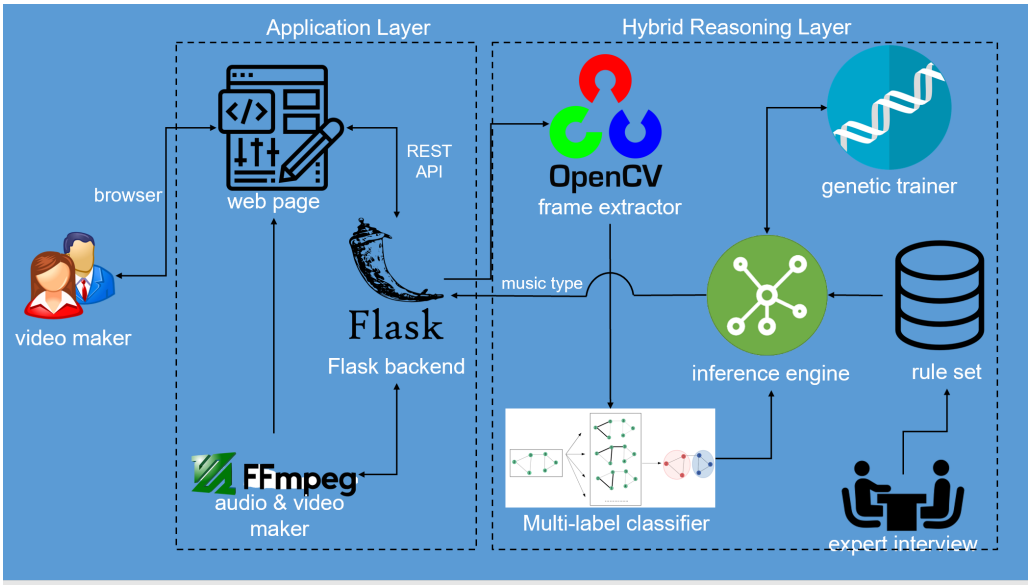


Figure 3: System architecture

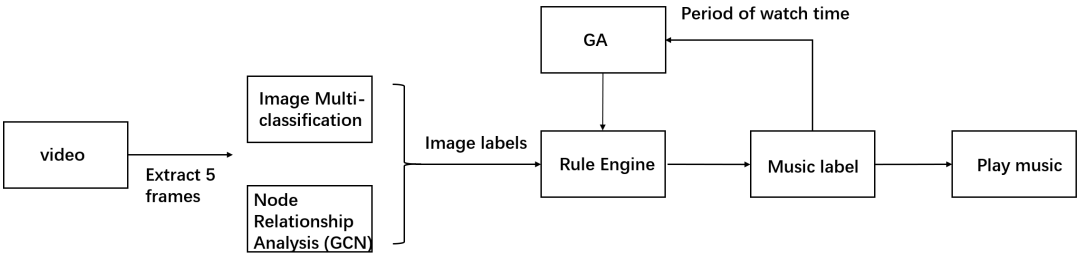


Figure 4: flow graph of Automatically Recommend Music System

image preprocessing (resize, rescale, gray) and we use Chebyshev Polynomials to approximate high level graph of input graph.

2.1.1. Graph Convolutional Networks(GCN)

The power of deep learning models lies in enabling automatic discovery of latent or abstract higher-level information from high dimensional neuroimaging data, which can be an important step to understand complex mental disorders. However, Convolutional neural networks (CNNs) do not generalize to irregular graphs since discretized convolutions are only defined for regular domains. Therefore, we use the spectral approach which provides a well-defined localization operator on graphs to define graph convolutions.

Graph convolutional neural networks (GCN) aim to extend the data representation and classification capabilities of convolutional neural networks, which are highly effective for signals defined on regular Euclidean domains, e.g. image and audio signals, to irregular, graph-structured data defined on non-Euclidean domains. The graph convolution is employed directly on graph structured data to extract highly meaningful patterns and features in the space domain. Formally, we are given the adjacency matrix $\boldsymbol{\mathcal{A}} \in \mathbb{R}^{n \times n}$. GCN is stacked by several convolutional layers and a single convolutional layer can be written as:

$$\boldsymbol{E}^{(l+1)} = \mathbf{ReLu}(\tilde{\boldsymbol{D}}^{-1/2}\hat{\boldsymbol{\mathcal{A}}}\tilde{\boldsymbol{D}}^{-1/2}\boldsymbol{E}^{(l)}\boldsymbol{W}^{(l)}), \tag{1}$$

where $\hat{\boldsymbol{\mathcal{A}}} = \boldsymbol{\mathcal{A}} + \boldsymbol{I}_n$, $\tilde{\boldsymbol{D}}_{ii} = \sum_j \hat{\boldsymbol{\mathcal{A}}}_{ij}$, $\boldsymbol{W}$ is a trainable weight matrix, $\boldsymbol{E}^{(l+1)}$ are the node embeddings (i.e., "messages") computed after l steps of the GCN, and the node embeddings $\boldsymbol{E}^{(l)}$ generated from the previous message-passing step.

GCN can be considered as a Laplacian smoothing operator for node features over graph structures. The architecture of GCN consists of a series of convolutional layers, each followed by Rectified Linear Unit (ReLU) activation functions to increase non-linearity. The first hidden layer $\boldsymbol{E}^{(0)}$ is set ot the input original node features. All layers share the same adjacency matrix. A full GCN run L iterations of Equation (1) to generate the final output node embeddings, $\boldsymbol{E}^{(L)}$.

To localize the filter and reduce the number of parameters, we employ the Chebyshev Polynomials to approximate the convoluational kernels. The computational complexity can be reduced with K-localized convolutions through the polynomial approximation [2]. By recursively applying a stack of graph convolutions with the 1st-order approximation, K-localized convolutions is computed to exploit the information from the K-order neighborhood of central nodes.

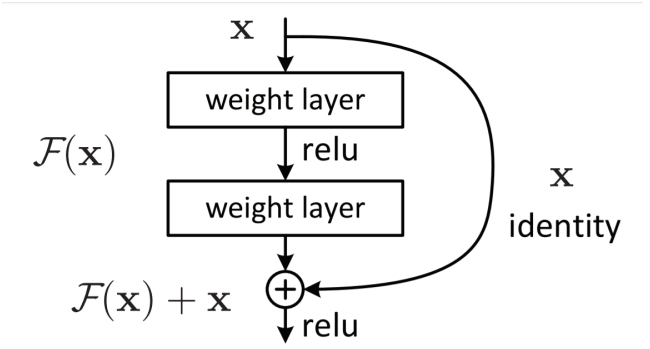


Figure 5: one block of resnet model

2.1.2. Resnet

Deeper neural networks are very difficult to train. We use Resnet [3] model to present a residual learning framework to ease the training of networks that are substantially deeper than those used previously. We explicitly reformulate the layers as learning residual functions with reference to the layer inputs, instead of learning unreferenced functions. One specific block of resnet model is shown in Figure.5.

In our system, we use resnet-101 for our model backbone.

2.1.3. ensemble of GCN and resnet

In this ensemble learning, we use 'Learning Differential Diagnosis of Skin Conditions with Co-occurrence Supervision using Graph Convolutional Networks' [4], which published by one of our team members to make entire model. The model overview is shown in Figure.6. We add GCN model on resnet model, at the end of the model, we dot multiplication between the GCN feature matrix and resnet embedding features. As said above, CNN have strong ability to process images which have translation invariance attribute, GCN have novel characteristic to process the un-euclidean structure, so our model ensemble two these sub-models can both take advantage of CNN and GCN, which can have a higher classification performance.

2.2. Rule engine

The rule engine model is the core of recommendation model, the input of rule engine is the image labels and the output is the specific music type. We obtain data base through interviews with relevant experts (teacher in piano and guitar shop), the specific interview dialogue is written in appendice.

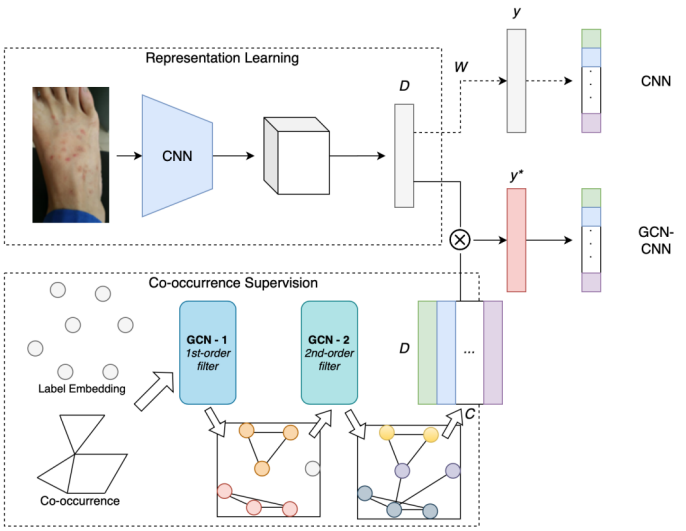


Figure 6: the GCN branch propagates label co-occurrence and semantic embedding. A trainable representation network has its feature vectors dot product with GCN output and generate final predictions.

In our knowledge base, we divide 80 categories in COCO database into 5 main music types (passion, excited, happy, relaxed, quiet) and two subtypes (static, active). The music we provide is divided into 5 categories, each of which has three categories representing the degree of each music type (high, middle, light)

We write our rules in three blocks in one txt file. The first block denotes the output label we choose which decides the main direction of next analysis, the second block decides the music type corresponding to the first label, the third block decides the music degree corresponding to second label. For the input label level, if the input image label is none, we just random select one music type and degree, if the input image label is one, we select one music type corresponding to this label, and the degree is middle, if the input image label is two or above two, we use the output possibility of each label to sort the output label, and select first two labels, the first label decides which music type to choose and the second one denotes the music degree. The degree mechanism denotes whether the subtype of the second label is same with the first ones, if same, rule engine will output the corresponding music type with high degree, in contrast, if not same, rule engine will output the corresponding music type with light degree. The pipeline of rule engine is shown in Figure.7.

In conclusion, the more labels input, the more specific music type we can recommend. However, this model also has some drawbacks in the recommendation

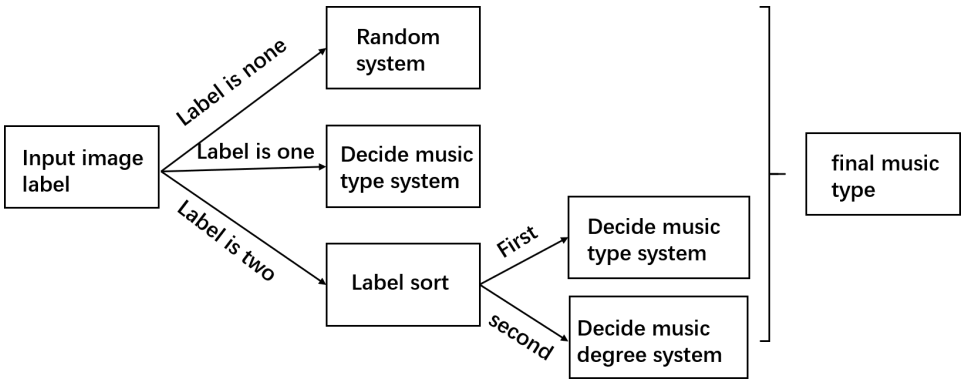


Figure 7: The pipeline of rule engine.

mechanism. For example, we cannot automatically update the recommended formula, and we cannot specifically learn the relationship between two input image labels. To deal with this problem, we use GA method to update the model.

2.3. GA

Based on our expert knowledge, we have built a rule engine to infer the music type forwardly based on the labels of the video. However, there might some gap and misunderstanding within the process of conducting the engine. Therefore, we apply Genetic Algorithm (GA) to help improve the structure of the engine. Generally, GA are composed of several parts: population, chromosome, fitness value, stopping criteria, parents selection, crossover and mutation The basic process of GA is showed as following Figure.8

2.3.1. Population Chromosome

In our project, we add another change array for inferring music type after obtaining the music type:

$$MusicType = abs((MainLabel + SubLabel) - ChangeArray) \quad (2)$$

where $MainLabel$ represents the label that occurs most in the video, $SubLabel$ represents the following one (if applicable) and the $ChangeArray$ is an additional array that is used to change the final output arbitrarily. To be more specific, the change array is a 10x1 vector where each value is an integer for changing the type. The dimension is 10 since the image label has 5 types so all the possible

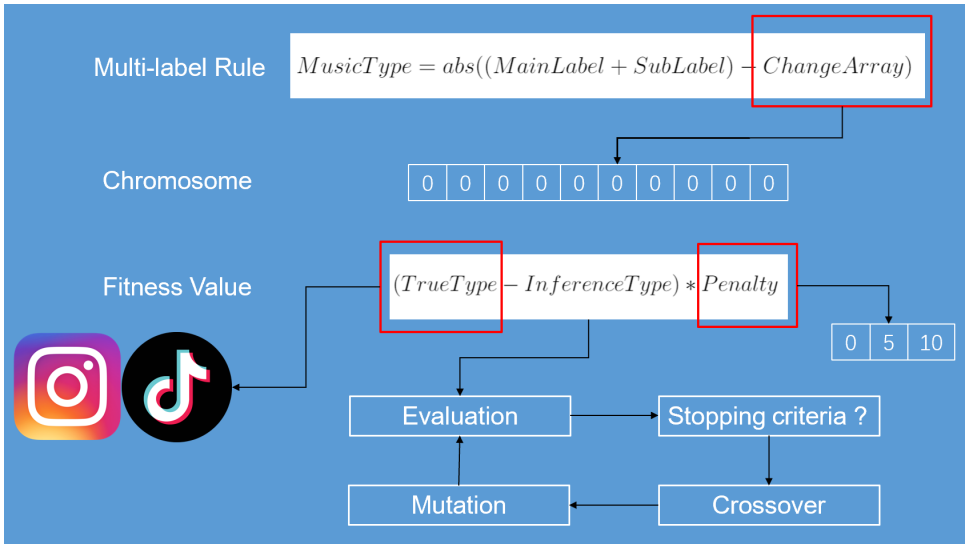


Figure 8: The pipeline of GA.

situations for two image labels are (5*4)/2=10. As explained above, one chromosome of our GA is the $ChangeArray$ and we design our population to have 5 chromosomes. To initial our first population, we set the array to identical within each chromosome and the five values are 0,1,4,6 and 7 correspondingly where 0 represents the original solution and the larger the value the more changes to the original one.

2.3.2. Fitness Value

The next step is define an efficient fitness value for us to evaluate each chromosome. Considering the fact that the distance between different music types is meaningful (eg, the distance between excited and passion is much smaller than that between excited and quiet), we define our music type from 0 to 9 where 0 represents the most passion type and 9 represents the most quiet one. Specifically, 0 presents most passion, 1 represents minor passion and so on. Then our fitness value can be calculated as follows:

$$FitnessValue = (TrueType - InferenceType) * Penalty \tag{3}$$

where the true type and the inference type are the 1 to 10 values defined above and the penalty depends on the inferred type and true type. If the inferred output has the same main type with the true type (eg, most excited and minor excited), then the penalty is 1. If the two types are all active or static (eg, excited and

passion or quiet and relaxed), then the penalty is 5. If the two are inverse types (eg, passion and relaxed), then the penalty is 10. The idea here is to penalize the chromosome more if the inference is completely wrong or partially wrong so that GA can converge faster. However, to make the calculation of fitness value complete we still need to obtain $TrueType$. To do that we utilize popular short video apps to get 100 high-likes video in the five music types (each type with 20).

2.3.3. Stopping Criteria

Common ways of stopping the GA including setting maximum iteration, the fitness value of one chromosome reaches the threshold or all the chromosomes in the population have become the same. In our project, we set the max iteration to be the number of videos since the GA will iterate once after processing one video. Besides, the threshold is 0 which means we require the rule engine to be accurate.

2.3.4. Parents Selection

In each iteration, if the solution cannot satisfy the requirement of certain criteria, then each chromosome will be selected according to the fitness value to become the parents of next generation. Due to the definition of the fitness value which represents the distance of inferred type and target type, the individuals with small fitness values will be selected and those with huge distances means a poor performance thus will be discarded.

The selection of parents is crucial to the convergence speed of GA and the performance of offspring. If the parents are selected poorly, then it is hard for GA to get a satisfying solution and the time cost can also be unacceptable. However, on the other hand, However, care should be taken to prevent an extremely suitable solution from taking over the entire population in several generations, as this leads to solutions that are close to each other in the solution space, leading to a loss of diversity.

Therefore, Maintaining population diversity is critical to the success of GA. Traditional ways of parents selection includes roulette wheel, stochastic universal sampling, tournament selection, rank selection method and random selection. In order to avoid the premature convergence and late convergence, the selection method should keep the randomness of one population while ensure the evolution of population. Considering these factors, we choose tournament selection as our method. Specifically, we utilize N-tournament method where N equals 3. The selection process is shown in Figure.9

In the N-tournament selection (N=3), we choose N chromosomes randomly and choose the one with smallest fitness value. Repeat this process 5 time until a

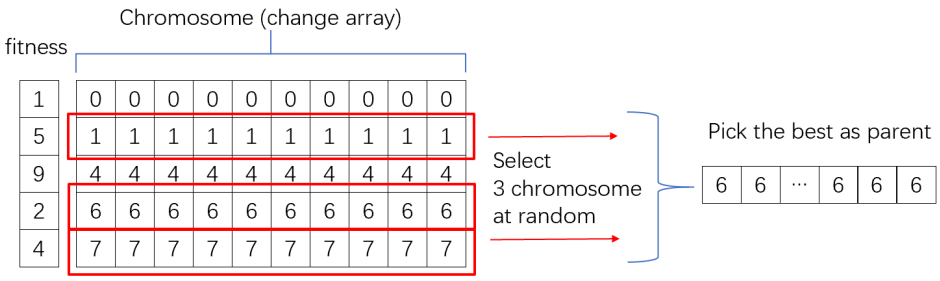


Figure 9: N-tournament(N=3) selection for parents.

new population is fulfilled. In such a way not only gives us opportunity to choose random chromosome but also exposes more to those better parents with smaller fitness value.

2.3.5. Crossover and Mutation

As mentioned above, one key element for GA's success is to maintain certain degree of randomness and prevent a good solution taking over the generation. Besides, the direction of evolution is also critical to seek for a good solution in the searching space. This is why the crossover and mutation parts are added in the process. Crossover often happens between two parents where their certain segments are cut, sliced and swapped to produce two offspring. Common ways of realizing crossover operation are single-point crossover, multiple-point crossover, uniform crossover, Davis'Order Crossover (OX1), etc. In our project we choose multiple-point crossover where the number of crossover points is fixed at 2. Another key element in the crossover is the crossover rate Pc which gives the expected number of chromosomes to undergo the crossover operation. We set the rate to be 0.75. The crossover process has the following steps:

1) Generate a random number r and compare it with Pc
2) If r $> Pc$, select two chromosomes for crossover;
3) For each pair of chromosome we generate two random numbers from the range 1 to 9 representing the two crossover points in each chromosome;
4) Swap the sliced part in the pair to produce new offspring

After crossover, the final step of a GA process is to mutate some of the chromosomes to introduce diversity of genetic populations like the real world. There are

several mutation methods: bit flip mutation, random reset, exchange mutation, scramble mutation, reverse mutation. We select random reset as our mutation method since we have a pool of possible values for each mutation bit (0-9). Like the crossover rate Pc, the mutation also has a mutation rate Pm to give the expected number of mutated bits. However, the difference is that Pm gives a low probability while Pc is high. In the mutation part we set Pm to 0.1. The mutation process is similar to the crossover:

1) For each bit generate a random number r and compare it with Pm;

2) If r $>Pm$, mutate the bit with value range from 0 to 9;

2.3.6. GA Training

We implemented our GA using the process mentioned above and trained it with 100 high-like short videos (selected on tiktok) in the corresponding 5 music types. After each iteration the rule engine will be improved by the GA until the stopping criteria. After iterations our population becomes the same and all scores a zero fitness value. The final change array is [0,2,1,9,6,1,1,2,2,0]. As the result shows, most of the music type changes little except for 3 (excited) and 4 (happy). The original excited type is changed to relaxed and the happy type is changed to excited. The small change in most type may be caused by the subjective bias when selecting the video type while the reason of huge gap in the change is because of limited number of our dataset (only 100 videos) and the rough mapping of image labels (80 labels mapping to 5 types).

2.4. FE-BE Design

We use Flask framework in our project. There are a number of frameworks for Python, including Flask, Tornado, Pyramid, and Django. Compare with other frameworks, Flask is more flexible, simple, low development cost, and it still have a basic MTV (Model-View-Template) structure.

In our root directory, we have a templates folder that stores a series of HTML template. The FE control cooperates with JS to store the target video in static folder, BE analyzes and extract 5 video frames, calculates the most suitable music type for the video and merges the target video and music. At last, the FE js controls and reads the result. At the same time ,FE JS will calculate the time user watches video, and return it to our GA for iteration.

Apart from what I have mention aboved, our FE also have some beautiful UI components, CSS.They can improve our user experience. The demo is shown in

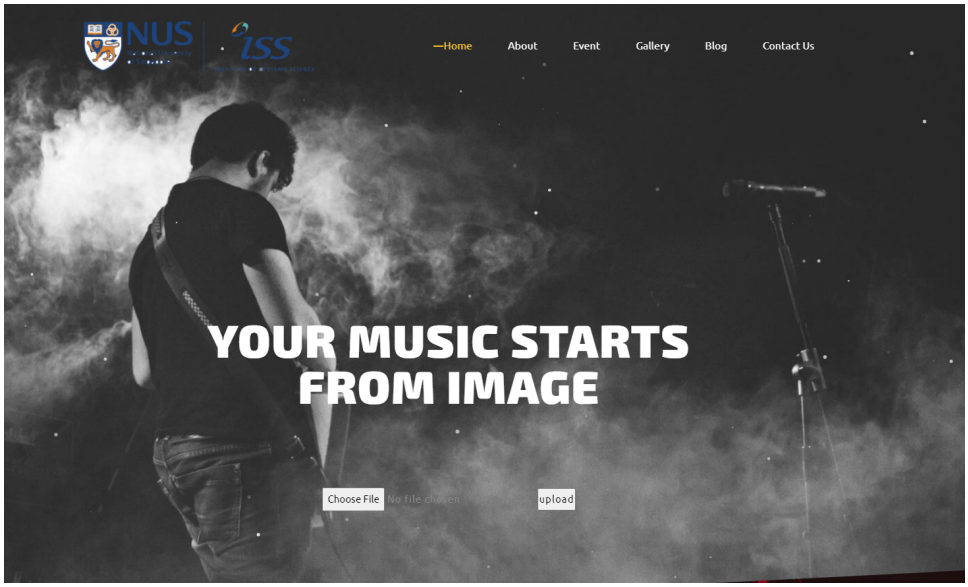


Figure 10: project demo 1

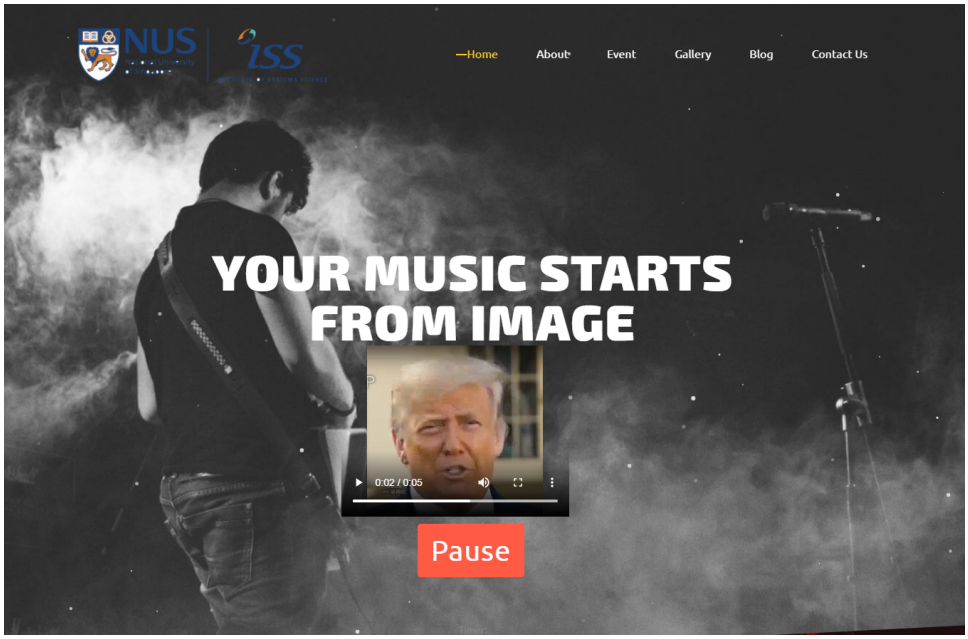


Figure 11: project demo 2

Figure.10 and Figure.11

3. APPENDIX

3.1. APPENDICES 1:Project proposal

3.1.1. What we do

In this project, we designed a system that it can automatically matches video and music. Among them, the video is uploaded by the user, and the music is already labeled and stored in our music library. The front end of our application is a web page, the framework uses flask of python, and the back end contains a series of functions such as GCN, GA and video and audio cutting.

3.1.2. Why we do it

In people's daily fragmented time,Short-video software is very popular because of its Social attributes. In this field, ML/DL technology also shows great potential, for example , the video recommendation system.

However,in our market survey and our own experience , we think in music recommendation part, AI technology has not been widely used.When a user is preparing to upload a video, the recommended music may only be based on the user's own attributes, such as age, gender, or the most popular music of the month or the year which may not suitable for the video, because they cannot associate the music with the video content.

We hope that we can use the video content information and the knowledge we learned in class , such as the GA , rule engine , and CNN to help short video software improve the music recommendation function

3.1.3. How we do it

In our application, when user upload a video, we will extract 5 frame as 5 image. Then , this image will process by our first model, which an ensemble learning of GCN and resnet. Then , we will get an output label for all 5 images, it will be the input of our rule engine model. Our rule engine model will help to match image label with music label. Then , our BE function will combine original video with our match-est music, generate output video to FE. Our GA will improve the structure of rule engine offline before the use of inference engine.

3.2. APPENDICES 2:Survey For Expert Knowledge

A short video maker needs to pick up a proper, unique and impressive song for their videos. However, with the development of network infrastructure, video makers have to speed up in making videos while maintaining the quality. To understand how they pick up their music according to the content, we surveyed a person who works as a self-media operation. The following is a conclusion from our survey:

We: What is the first step in picking up the music for a certain music type?

Expert: Usually I will firstly ensure the theme of my video before making it. The theme can comes from my inspiration, messages and questions from audience or inspired by other hot videos. The first time to pick a theme can be slow but after some time I form my own theme library. Now if I decide to make a video, I will check my library firstly and see if there is one satisfying my need.

We: So basically the music type relates to the theme of the video?

Expert: Exactly. Once I choose the theme, a brain storm is necessary to help me record all possible elements that might be used in the video. For example. if I choose to make a video focusing on teaching audience to take photos naturally, I may write down possible scenes that audience might want to take photos such as babies playing with pets or a quiet and nice scenery. Then I will take them as teaching examples in my video.

We: Will these examples produced in the brain storm change the music type?

Expert: Sometimes it will. However, the basic type will not change but the degree can vary from light to extreme. However, nowadays there are some hot videos that impresses the audience by contradicted music type.

We: And some of your videos will do that too, right?

Expert: Currently no, maybe I will consider that in the future.

We: You mention basic music type just now. Could you tell us how many basic types to choose from?

Expert: The two basic types of my videos actually refers to active and static. If I choose to make an active one, then I will not consider static type and vice versa.

We: If given you a video, how would you decide it is an active or static one?

Expert: That's a good question. Usually I will overview the video quickly by just looking at several seconds. Most of videos can be identified in this part since the content of a video is expanded according to the theme. By sampling several seconds the tune of the video is clear. Besides, a good video maker will have a structured and informative transcript to help him. Taking my videos as an example, I will follow a introduction-supportive details-conclusion structure in

my transcript. Therefore, if you want to check the theme of a video, you can focus on the beginning and the conclusion part which gives clues to the theme.

We:So we could determine the basic type by checking certain parts of the video?

Expert: Yes.

We: And what music types will you label as active and what will be static?

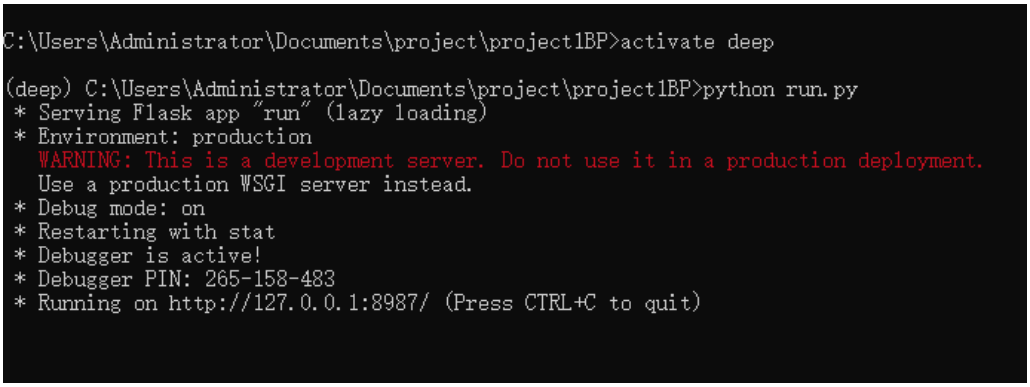
Expert: It is hard to tell generally since every self-media practitioner has their own scope of videos. What I can say is in common situations the passion, excited and happy music will be used in active video and relaxed and quiet one will be in static. But the boundaries between them sometimes can be confusing and subjective and I often don't differentiate them so much.

We: That's really helpful. We have already taken too much for your time. Thanks for your answers. Hope you can make a hit video in the future.

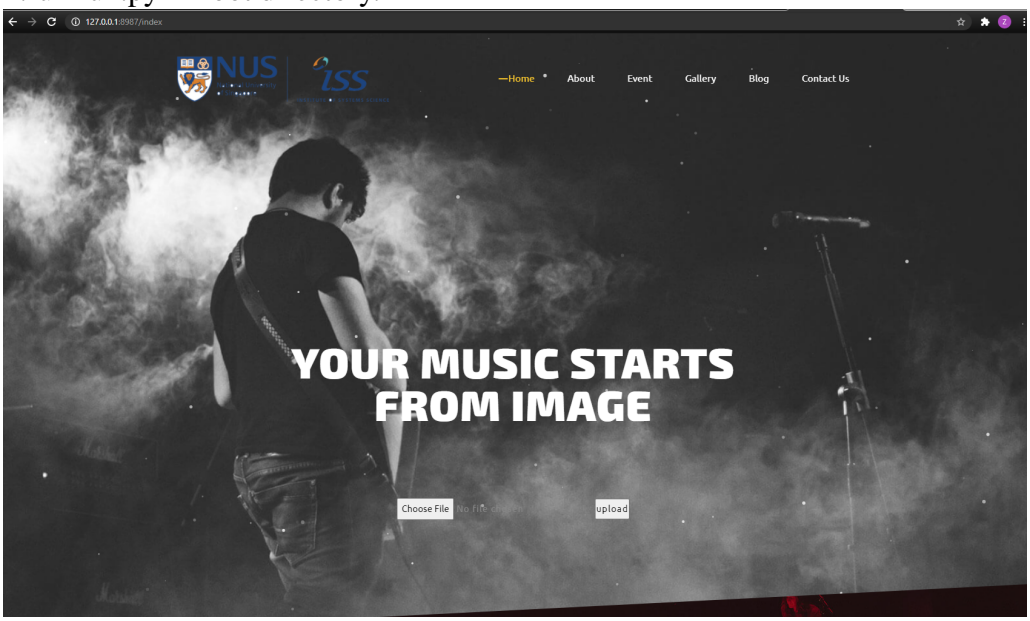
3.3. APPENDICES 3:Installation

1.Download project from address
2.Download Coco checkpoint from address,
Put it in to project root directory.
3.Install ffmpeg. For more detail , please go to https://ffmpeg.org/
4.Make sure torchnet >= 3.0.3
5.Make sure flask >= 1.1.1
6.Make sure OpenCV >= 4.4.0
7.Make sure torchvision >= 0.7.0
8.Video introduction link: address

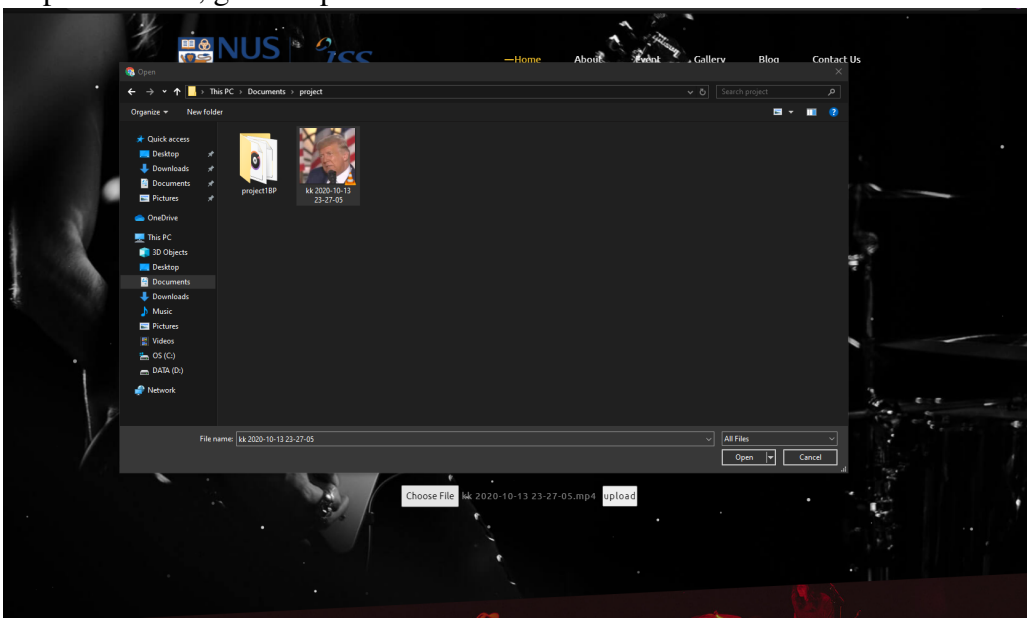
3.4. APPENDICES 4: User Guide



1.run run.py in root directory.

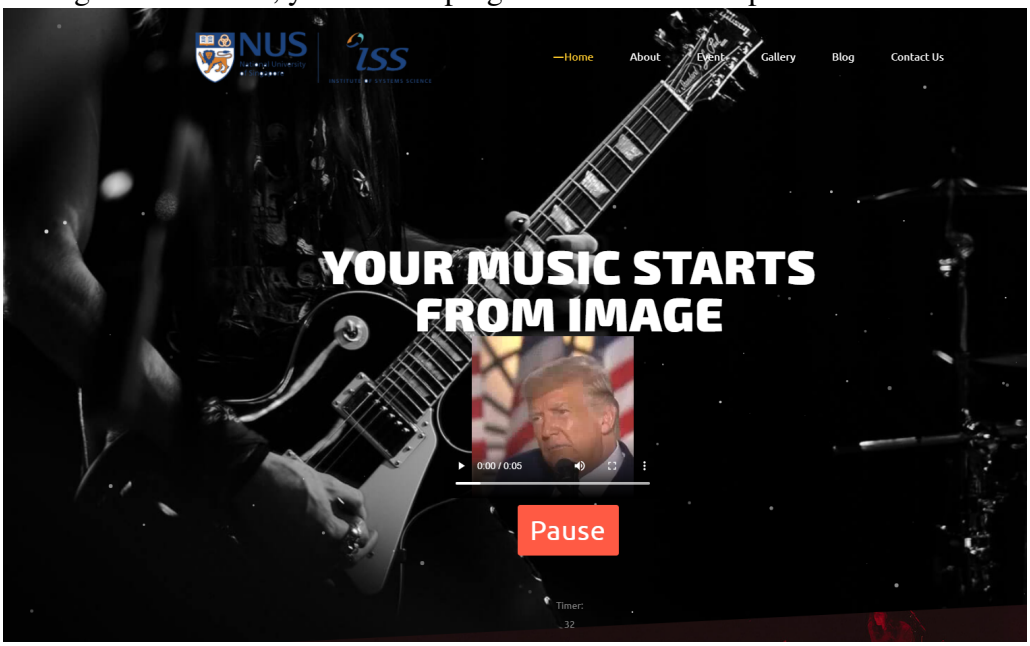


2.open browser, go to http://127.0.0.1:8987/index.



3.Click choose file button, select a MP4 or AVI format file and upload it.(we

prepare "$demo_video/demo_video.mp4$" for you to test))
4. Waiting for 1 - 5 mins, you can see progress in terminal output



5. return and play a new a video in FE ,

3.5. APPENDICES 5:Mapped System Functionalities

Our project has mainly four parts: GCN and Resnet for multi-label image classification, forward rule engine for inferring music type, genetic algorithm for improving the structure of rule engine and flask package for user interface. Therefore, GCN and Resnet maps the concept mentioned in CGS, rule engine is the technique learned in MR and genetic algorithm is one of the important search algorithms studied in RS.

3.6. APPENDICES 6:JIANG HAO Personal report

3.6.1. personal contribution to group project

In this project, I am responsible for doing multi-label classification model[1] and rule engine. Specifically, in the multi-label classification part, I try to change the original data preprocessing from Image library to opencv, which we can get a more stable and robust result. Besides, I use Chebyshev Polynomials to approximate the high level graph, and the specific method is in [4], which I published in 2020-MICCAI.

In the rule engine part, Zhang JiuYun and me wrote the rules (extract by knowledge base) into one txt file, then I write a txt reader in python to achieve the model. Specifically, using the rules, this model can return a music when input music labels, search the target music in our music fold and return the music to UI. Besides, in the training part, rule engine can combine with GA model to update the rule base. We work together to stitch each model together.

Besides, we also do some task together (achieve the model, debug and fine tune), like frame extracting of video and model connection debugging.

3.6.2. What I've learnt

First, I knew the different from academic algorithm and real industrial technology, academic aims to create a new innovative model, but the real landed technology needs to consider more metrics such as speed and money not only how novel the model is or the high accuracy. Academics are the forerunners of technology, and the technology acquired by enterprises can truly create value.

Second, through this project, I have a more comprehensive and profound understanding of rule engine, Rule engines allow me to say "What to do" not "How to do it". The key advantage of this point is that using rules can make it easy to express solutions to difficult problems and consequently have those solutions verified (rules are much easier to read then code).

Finally, with cooperate with other two strong teammates, I also learnt the knowledge of how to make UI and GA model.

3.6.3. Skills in other situation

In the image multi-label classification part, we can use it in smart driving field, which can use sensor to and camera to do the image classification, then we can use further image detection method like faster-RCNN model to do the two stage image detection task.

In the rule engine field, we can use it in business rule and data analyse because rule engine is very flexible to use. For example, we can use rule engine into

opening time and opening hours of airport gates, due to different time has different passenger flow, proper airport opening hours and length of time can strike a balance between airport safety and number of flying population. This is more important because the covid-19 and it can be more safety if there are relatively fewer people in airport.

3.7. APPENDICES 7:MA ZEYU Personal report

3.7.1. personal contribution (MA ZEYU)

For this project, I am responsible for the front-end and back-end design.After Model part finish(including GCN and GA), I put them into back-end and combination them with our Flask model.In front-end part, I use some javascript to do a timing function, which provide data for our GA iteration.I also complete video editing, audio and audio synthesis, video and audio synthesis part, which is another main function for our project.

I think we are a team with a clear division of labor. In the beginning ,we have a discussion, each of us participated in and we decide which model we should use, how to use the knowledge in the class to complete our project.

3.7.2. What I learn (MA ZEYU)

Sometimes, although we think ideas are good, when we actually apply them, we will encounter a series of difficulties. At the beginning we try to do a combination of static pictures and music, but this would cause the user experience to be not good enough. Finally, we adopted the strategy of combining video and a new music. We also used genetic algorithms to improve the model. At the beginning, we wanted to use a neural network to match image labels with music, but we didn't have many such dataset, so in the end we used a knowledge engine for matching.

Secondly, when we combine class knowledge and models, we often stack models. Although it looks very complicated, some models are not necessary at all, and we need to consider the FE and BE response speed. Our final model is the result of our discussion, and we consider the most efficient structure.

Finally, I think teamwork is very important. We can learn a lot from the team members, or they can supplement or improve our ideas, and use a better structure to accomplish our goals.I am very proud to be able to team up with good people, and compare with other part , I learned the most from them.

3.7.3. Skills in other situation

In this process, we have used many technologies which we learn from class, such as GA, GCN, and knowledge engine. Among them, when we want to get some mutations, make some changes to our algorithm, or iterate multiple times, genetic algorithms are very suitable.

GCN is a very new and interesting model, I learned a lot from the team members.It has great potential in processing text, mult-label info, or even DNA and protein in biology (this is the internship I will participate in next year). Because

they are not like pictures that can be represented by a two-dimensional matrix, these applications can only be represented by graph, which are more complex structures.

For the knowledge engine,when it is difficult for us to express conditions with if-else, the rule engine is very useful. if we want to build a cargo distribution system, frequently changing rules, or risk control , rule engine is a good choice.

3.8. APPENDICES 8:ZHANG JIUYUN Personal Report

3.8.1. Personal Contribution

Our project is a web application for video makers. The backend is composed of the following part: frame extraction, image identification, rule engine and GA. I firstly designed the processing flow of our project and determined the techniques that we need. Then I transferred the expert knowledge into rules and designed the rule engine with flexibility for GA improvement with Jiang Hao. After that I defined corresponding definitions (chromosome, fitness value, etc.) for implementing GA and found 100 videos with teammates to train GA. Then I implement the GA step by step from fitness value evaluation to chromosome crossover and mutation. Finally I integrated the backend codes with frontend interface and ran through our application from input to final output to make sure there is no bugs within it.

3.8.2. What I Learned

I think my gains can be divided into mainly three parts: hybrid reasoning system, rule engine and GA. Firstly, the design of the a hybrid reasoning system should be conducted carefully. When I first time designed the system, I fixed the structure of the rule engine and found that GA can do nothing to the engine. Besides, the interaction between different parts within the hybrid system should be efficient. At first I decided to use the listening time of our output video as the fitness value. But it gave us a bad result as the GA converges slowly. So I defined another fitness value to change the rule engine's structure directly and it worked successfully. Secondly, as for the rule engine, I learned the difference between programming language's if-else and inference engine. In rule-based system we have rules floating around and they are independent from the engine. The engine's job is pick proper rules based on input data. However, in imperative language it is the programmer tell machine which rule to conduct first and which last. So coding in a rule-based system pays more attention to the if-then rules and their execution order and the rest will be handled by inference engine. Thirdly, I learned how to design and implement an efficient GA. I think the most important part in GA design is to define the chromosome and fitness values with clear meaning and purpose. Once they are determined, the rest part can be developed step by step easily. For the non-technical part,

3.8.3. Skills in Other Situation

We mainly applied three domains techniques: multi-label image classification, inference engine and genetic algorithm. Considering the image classification part,

it can be utilized with Internet of Things to make more terminal equipment smarter and more convenient. For example, face recognition could be added to the access control to support the safety check.

Inference engine is suitable in building business system which requires employee to check various business rules manually. With the support of rule engine, not only many human labors can be liberated but also adding a new rule become easy and convenient.

As for genetic algorithm, I think it can used to improve different machine learning models within different domains including rule engine, neural networks or linear regression. Besides, GA is also an efficient searching algorithm which focuses on finding global optimal solution. Therefore GA can also be used in planning situations such as course schedule planning or route planning.

Last but not the least, hybrid reasoning system which can be composed of various techniques including decision tree, rule engine, neural networks or GA has a flexible structure and performs better than the individual part thus is widely used in business decision, business forecast and software agent system.

References

[1] Z.-M. Chen, X.-S. Wei, P. Wang, Y. Guo, Multi-label image recognition with graph convolutional networks, in: Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition, 2019, pp. 5177–5186.

[2] M. Defferrard, X. Bresson, P. Vandergheynst, Convolutional neural networks on graphs with fast localized spectral filtering, in: Advances in neural information processing systems, 2016, pp. 3844–3852.

[3] K. He, X. Zhang, S. Ren, J. Sun, Deep residual learning for image recognition, in: Proceedings of the IEEE conference on computer vision and pattern recognition, 2016, pp. 770–778.

[4] J. Wu, H. Jiang, X. Ding, A. Konda, J. Han, Y. Zhang, Q. Li, Learning differential diagnosis of skin conditions with co-occurrence supervision using graph convolutional networks, in: International Conference on Medical Image Computing and Computer-Assisted Intervention, Springer, 2020, pp. 335–344.