My Exploration on Mechanistic Interpretability

Yueyan Li (Sirius)
almightygod007@163.com

Who am I?

Hi! I am now a postgraduate student and preparing to apply for a PhD after the first year of my master education (which is allowed in my college). I majored in Electronic Information Engineering during my undergraduate studies. My research now focuses on natural language processing (mainly human-machine dialogue in our lab), while I'm also interested in anything cool in machine learning and deep learning.

I gradually get bored with the mainstream research on LLMs (e.g. finetuning a pre-trained model or building a pipeline for agents) because I can't figure out what's going on in the model! I got a feeling of emptiness in winter last year so I made a decision to delve into the field of interpretability of neural networks.

I worked at Zhipu.AI (the team who develops the ChatGLM series) the first half of this year and read a lot of papers and posts related to mechanistic interpretability at the same time. I followed the research at Anthropic (transformer circuits thread) and gradually figured out what people are doing and how this field comes to the situation today. I was deeply motivated by the work done by Neel Nanda, Chris Olah, Arthur Conmy and all the related researchers in this field and decided that mech interp is what I want to study in the future.

What have I done in this google doc?

Though I've read a lot of papers in this field, I haven't done any mech interp research. So I followed the Arena 3.0 and SAELens tutorial and did something that I think is interesting and meaningful. Here's what I've done:

1. Explored the emotional and multilingual features using an SAE on GPT-2-small and found something cool to me (the main part of my project).
2. Trained SAEs on Phi-2 using SAELens and got confused by the curves (just as a try).
3. Noticed some bugs in TransformerLens and the Arena 3.0 tutorial and fixed it.

I did all the above in VSCode using GPUs in my lab. The code can be found in my repository: https://github.com/Siriuslala/saeExploration.git

I did experiments in jupyter notebooks, so you can just go through the multilingual_study.ipynb.

I've tried my best in the limited time though I think there's more I wanna do. I'm really looking forward to working with you all. Hope you like it!

1 Exploring Emotional and Multilingual Features In GPT2-Small

1.1 Exploring Emotional Features

I believe there are features that can reflect emotions in gpt2-small.

I think it's an easy start for me because I can check out the autoencoder features to find out the features that fire most actively.

Besides, I also want to do some exploration on the features among different layers and different positions of the model.

1.1.1 Find features that reflect positive emotions

To find the features related to a specific emotion and reduce data deviation, I write five sentences containing the key words for each emotion. For example, for happy emotions I have:

prompt_happy = ["I'll be on a vacation tomorrow and I'm so happy.",
"My mom brings home a new puppy and I'm so happy.",
"I'm so glad I got the job I wanted.",
"I feel so happy when I'm with my friends.",
"I'm so happy I got the promotion I wanted.",]

I choose to look for features that reflect happiness and sadness. Apart from that, I also wonder if the feature that reflects excitedness has something to do with the one that reflects happiness (they are alike from the semantic level at least.)

For a start, I inspected the residual stream in layer_7. The SAE we choose is **gpt2-small-res-jb** which hooks at the residual stream at the entrance of a layer.
The prompts were fed into the model and the outputs were SAE activations. I checked the activations at the word “happy” for all the prompts and calculated the mean value of them. I visualized them as below:

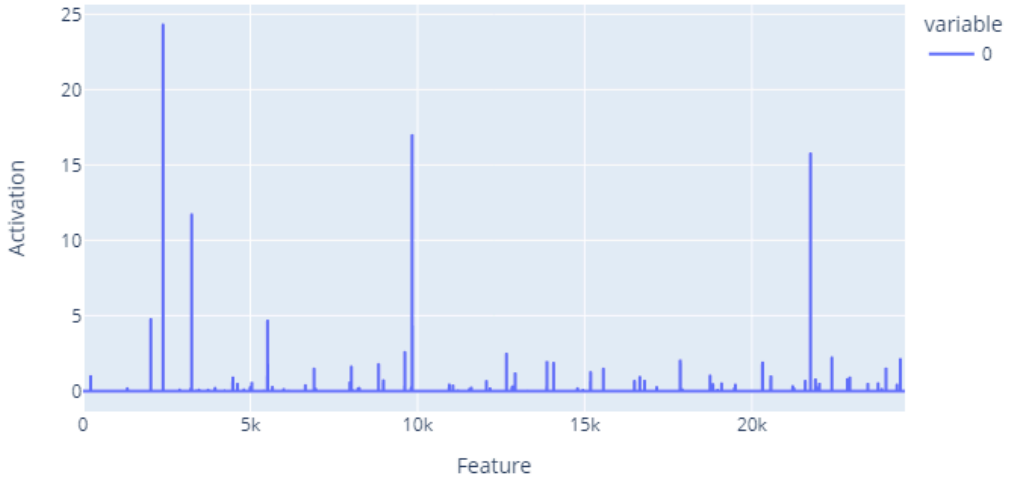

Fig 1: Feature activations on the happy emotion at the residual pre stream in layer_7.

Obviously there are three autoencoder features that activate most actively on the happy emotion, and their feature ids are 2392, 9840 and 21753. I checked the feature 2392 in Neuronpedia and got its feature dashboard:

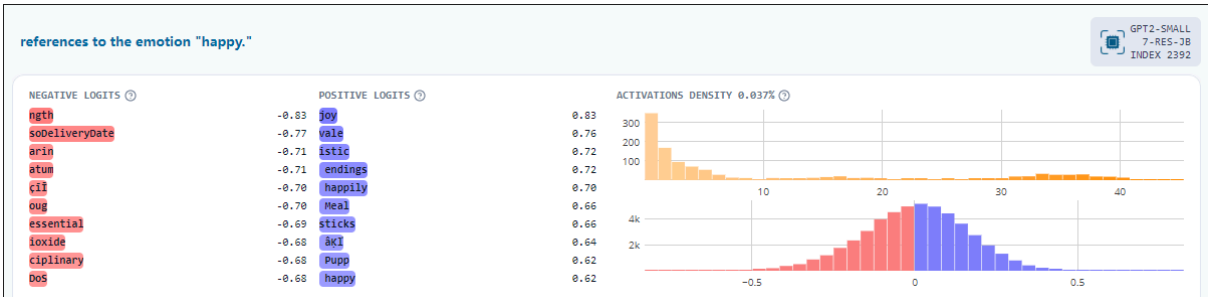

Fig 2: Feature activations on the happy emotion at the residual pre stream in layer_7.

Later I found features for sadness and excitedness in the same way. The top-3 features activating for “sad” and “excited” are [2045, 23774, 10866] and [8935, 9840, 3247] respectively.

1.1.2 Compare the features related to happiness and excitedness

I want to see the difference between "happy features" and "excited features" since they are both positive emotions. So I compared their top-3 features and only kept the features that activate on both “happy” and “excited”. They are visualized as below:

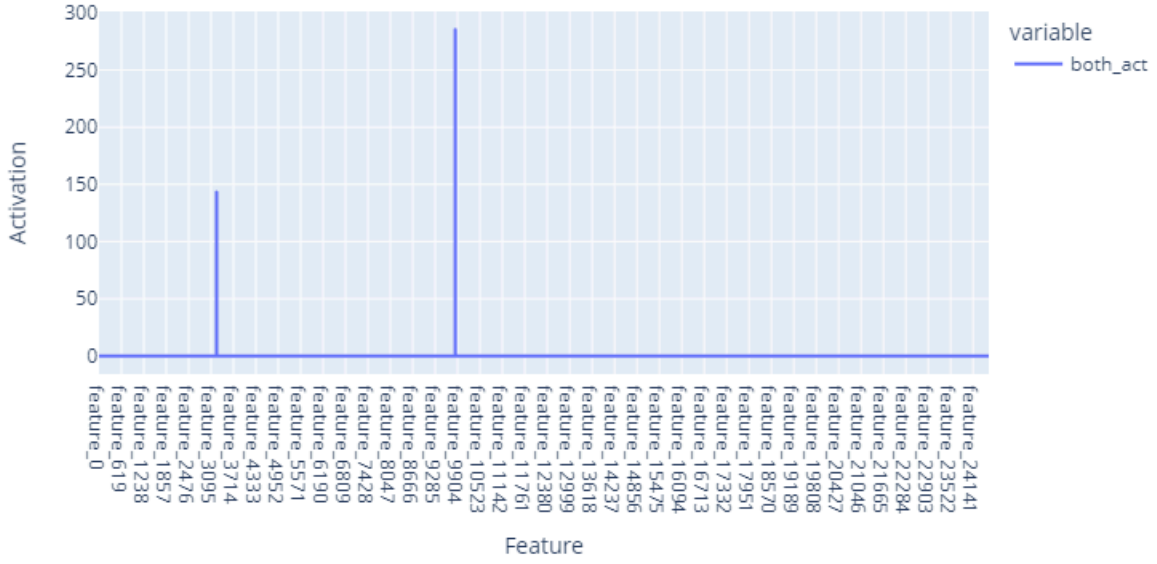

Fig 3: Feature activations on both happy and excited emotions at the residual pre stream in layer_7.

From the figure above, we can easily find out the two features shared by happiness and excitedness. It seems that these two features contain positive emotion concepts, which means they are close to each other on the semantic level.

1.1.3 A deeper investigation into the features related to happiness

From the previous result, we can see that there are 3 features that fire quite actively on happiness. Since they share similar semantic meanings, I guess that the representations of features activating on the same emotion (i.e. 2392, 9840 and 21753) have high similarities.

To prove this, I try to inspect the representation of one prompt which expresses happiness and calculate the cosine similarities among representations of different features. Here the representations of different layers are calculated as below:

feat_repr = W_dec * SAE_activations

Note that the * operation is a dot product. The W_dec in the formula above is the decoder matrix of SAE with a shape of **(d_sae, d_model)**. We know that according to the definition of dictionary learning, each vector in the dimension of d_sae corresponds to a base vector for an autoencoder feature. Each element in the SAE_activations can be seen as the intensity of the feature at that position. So by multiplying W_dec and SAE_activations we can get a representation of shape **(d_model,)** which expresses the features in the vector space of SAE that is sparser and more interpretable than that of the original model.

To visualize the similarities among features, I choose to use the heatmap. Note that in order to make it clear, I mainly focus on features 2392, 9840 and 21753 and using negative sampling to get some other

features for comparison. I randomly pick 6 features except for the three features mentioned above. The similarities are calculated and shown as below:

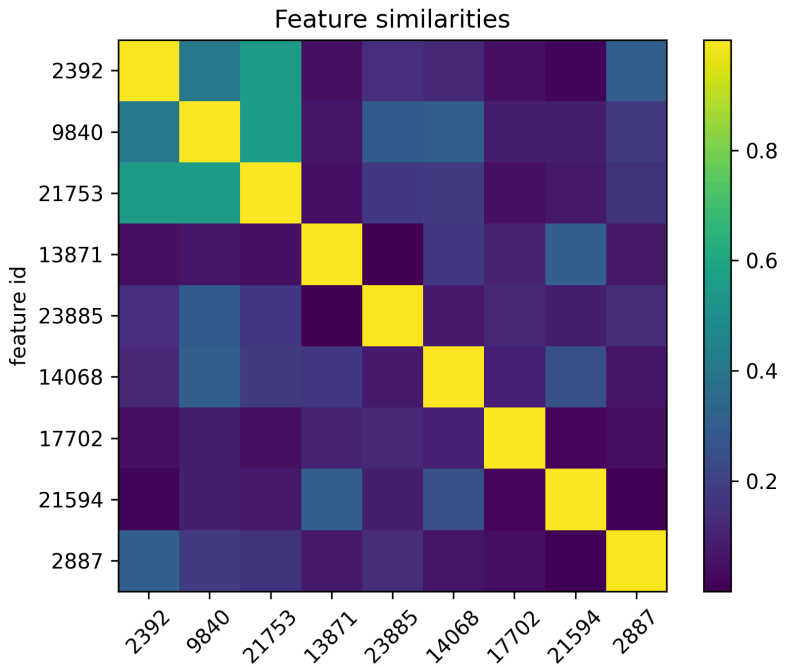

Fig 4: Feature similarities on the word “happy” at the residual pre stream in layer_7.

Obviously we can find that the three features that fire most actively on “happy” are more similar to each other (the top left 3*3 square), thus my hypothesis is proved intuitively.

1.1.4 Features in different layers

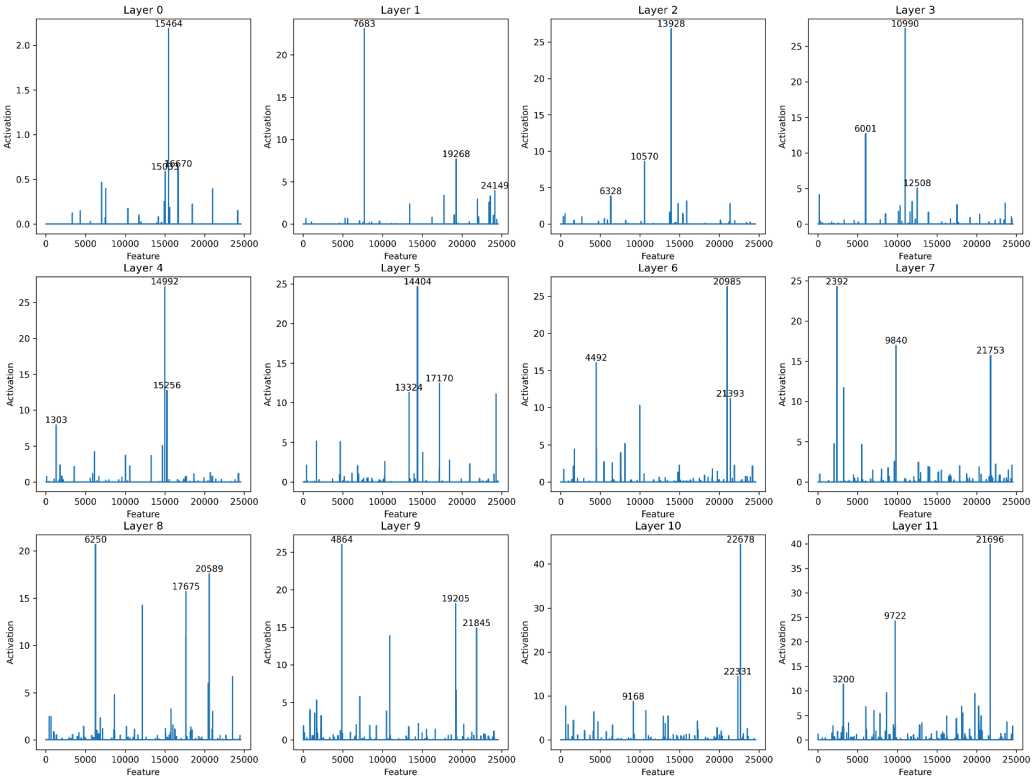

Fig 5: Feature activations on the word “happy” at the residual pre stream in 12 layers.

Previously I inspected in the 8th layer of gpt2-small and found some emotional features. Now I want to know if similar features exist in other layers, and how they are related to each other.
I inspect the autoencoder features in each layer and observe their activations on the word “happy”. The result is shown in Fig 5. We can find that:

- Each layer has more than 3 features that activate on the happy emotion.
- The positions of activating features are different from those in other layers.

Though the positions of activating features are different, I guess it's just the problem of feature orders. For example, the feature 7683 in layer_1 may be the same kind or even exactly the same feature as feature 13928 in layer_3, though the positions are different.
In order to prove this, I choose to visualize the feature representations of SAE outputs. The result is shown in Fig 6.

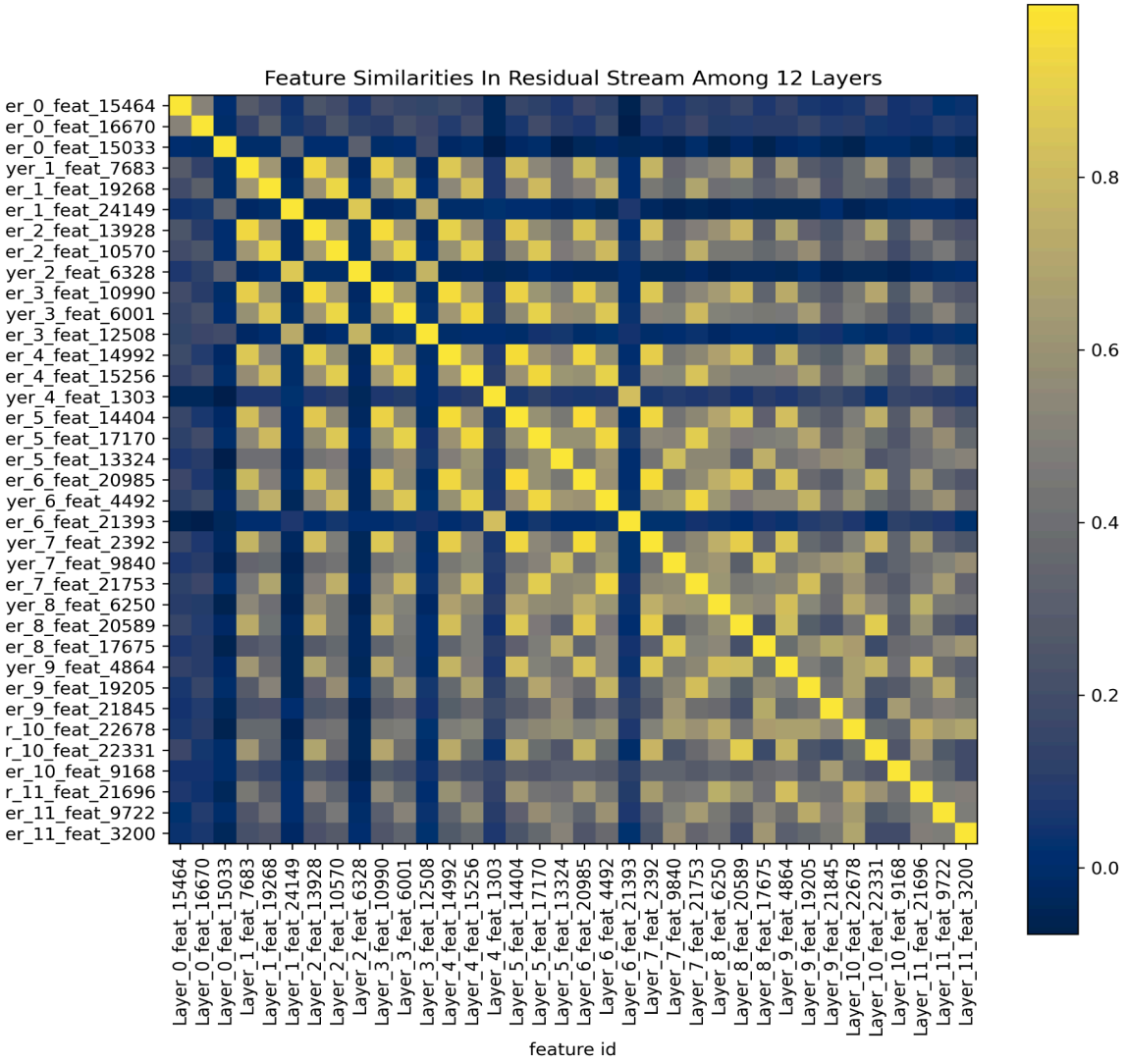

Fig 6: Feature similarities on the word “happy” at the residual pre stream in 12 layers.

From the figure above, I got some interesting points about the feature distribution:

- **The features in the first layer have lower similarities with those in later layers**. I guess this is because the first layer gets limited information from previous layers, so it cannot

express relatively complicated concepts like emotions. I think maybe the first layer contains some low level concepts that are not shown in this figure, which I will explore in the future.

- **A feature in a layer often corresponds to a feature in another layer**. For example, feature 7683 in layer_1 corresponds to feature 13928 in layer_2, which has a similarity of 0.939. This means they are related to each other across different layers, sharing similar semantic meanings.
- **A feature in a layer tends to be more alike to features in nearby layers**. For example, feature 7683 in layer_1 is more similar to feature 13928 in layer_2 than feature 2392 in layer_7, with a similarity of 0.939 to 0.825. I think it's because the vector spaces of nearby layers are relatively close to each other. When two layers are far from each other, the difference between their vector spaces is significant due to a lot of linear and nonlinear manipulations between layers. Thus the features would share low similarities regardless of similar semantic meanings.

1.2 Explore Multilingual features

I remember GPT2 can speak French because of the English-to-French translation ability shown in the paper "Language Models are Unsupervised Multitask Learners".
I'm a language lover so I want to find out the possible multilingual features in gpt2-small.

1.2.1 Emotion features in English and French

Firstly I test the French ability on gpt2-small. We can see that the generation of the word "heureux" requires 3 steps. Though the first token " he" is not generated with a high probability, the model successfully generates the next two tokens. This implies that gpt2-small has the knowledge of French.

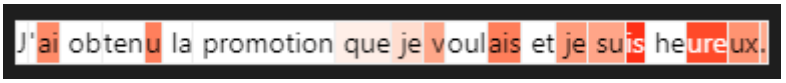

Fig 7: Probabilities of each token on a prompt in French.

Unlike English, French words are splitted by the tokenizer more often because French accounts for a smaller part than English in the training corpus for the tokenizer. So it's not that easy to use the way I study features in English before.
To deal with this, I examine the activations on all separated tokens in a French word, and add them together to find out the features that fire on it (weighted sum maybe better but for convenience I use the common sum).

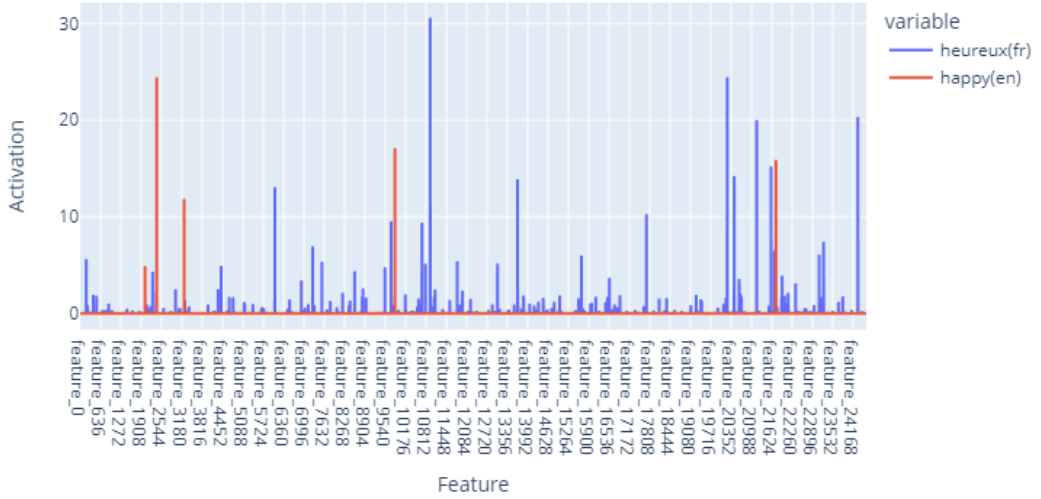

Fig 8: Features activating on French word 'heureux' and English word 'happy'.

As we can see in Fig 7, there exists some features activating for the French word, but they are not the same as those activating for the English word "happy".

1.2.2 Find a French emotional feature in attention layers

Inspired by the task of machine translation, attention layers attract my attention because attention layers have the ability to combine concepts together!

So speaking of the relations between two languages, e.g. English and French, I guess that most features which are related to French exist in attention layers. That is to say: instead of staying inside the MLP layers or the residual streams, the French features might be represented indirectly by the attention layers.

Another reason I think is that the training corpus contains less French than English, and the vocabulary of the tokenizer doesn't even contain the word "heureux"("happy" in English), thus the model cannot directly assign a feature to a concept in French.

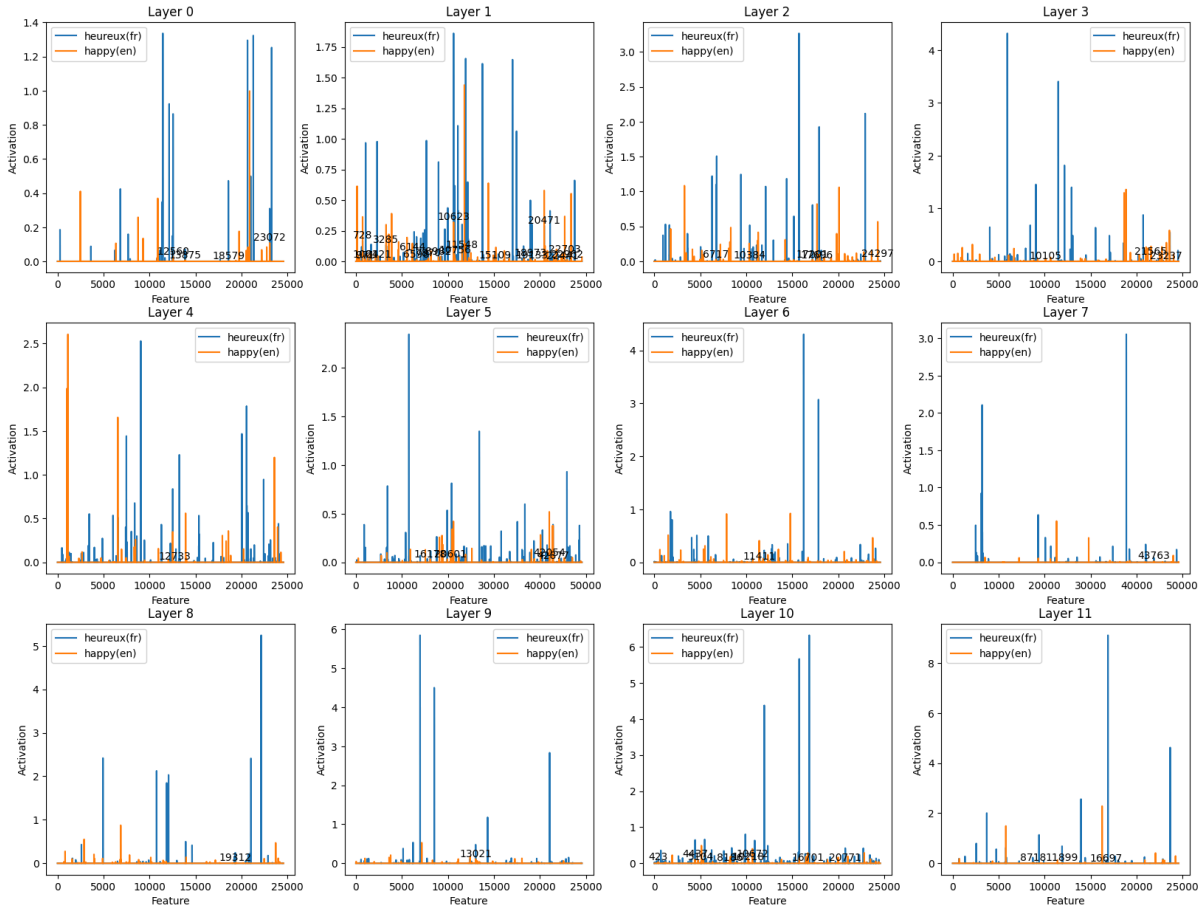

Fig 9: Feature activations at the keywords 'happy' and “heureux” in each attention layer.

The feature activations are shown in Fig 8. Unfortunately I didn’t find any features that fire actively on both “happy” and “heureux”, though there exist features that fire for “happy” or “heureux” alone.

Later I checked the activations in residual streams, and the situations are similar. Most features that fire on both English and French are features with extremely low activation values (see the bottom of each sub-fig). Maybe the features that combine English and French together do exist in these small activations, but I haven’t got a way to interpret them. If so, do the small features act alone or together or together with the features with remarkable activations? This issue remains to be solved.

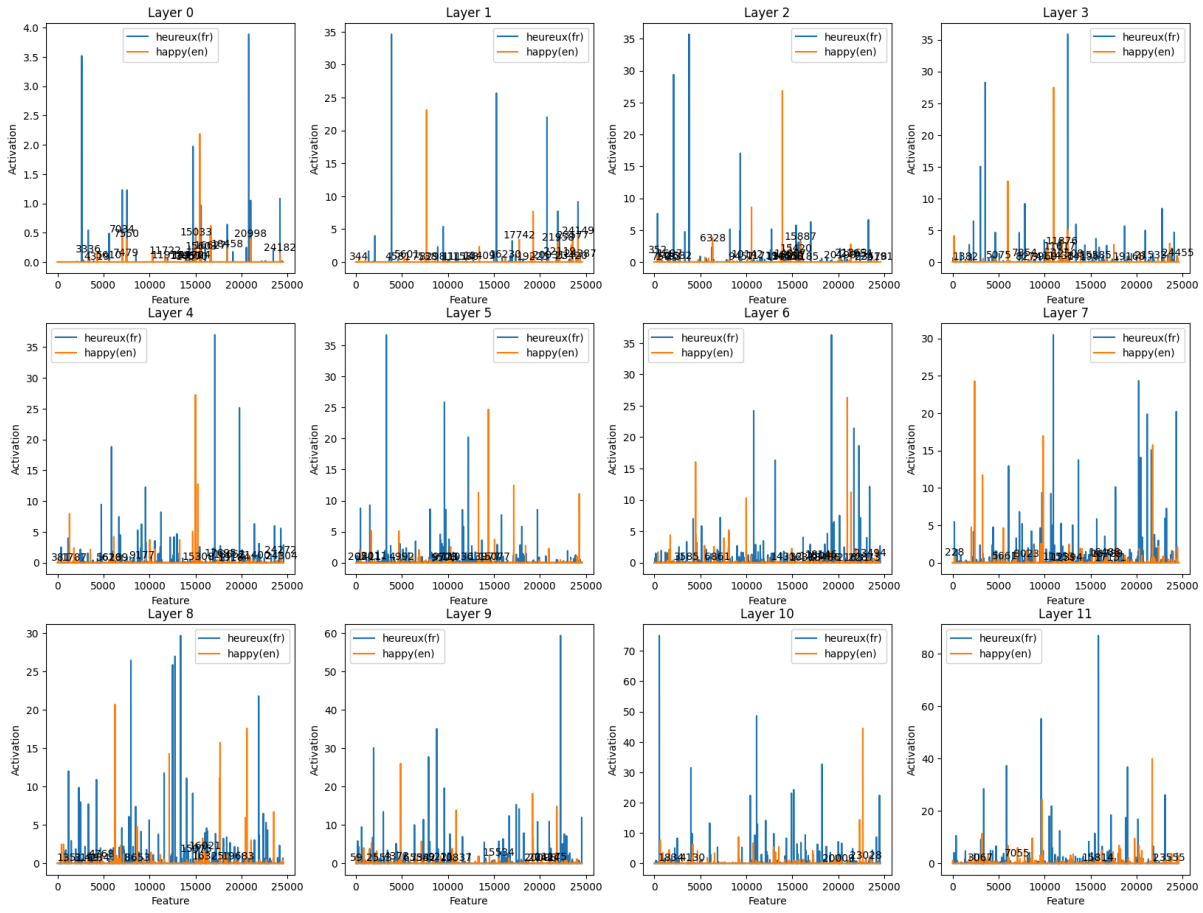

Fig 10: Feature activations at the keywords 'happy' and “heureux” in each attention layer.

From my perspective, there are two possible reasons why we cannot find features that could fire actively on both English and French:

- **The model we study is too small**. The capacity of gpt2-small is limited so it stores information in a much more subtle way (e.g. superposition) which is hard for us to understand.
- **The SAE doesn’t express the features we want**. This has something to do with the size of the SAE, the training strategy and training corpus…so it’s really a big issue to train a better SAE!
- **Some subtle relations between features that are not found due to my limited ability. -_-**

1.2.3 Steering a French prompt with English features

Though I failed to get the multilingual features I want, I believe that they must exist. So at the end of the exploration, I perform model steering with the emotion features I detected previously. Specifically, I try to guide the generation of gpt2-small to positive or negative styles. Surprisingly, **I find this work across different languages: a feature activating for “happy” can lead to positive content in French!** Below are some of my plays:

Prompt: "Today is Tuesday and I am"
Feature Vector: feature 2392 in residual stream pre of layer 7

Feature Property: happiness
Generations:

- Today is Tuesday and I am to have **achieved this success**. It means, however, that a second way
- Today is Tuesday and I am to **have completed this job thanks to** their eSoup thank you program.
- Today is Tuesday and I am to get some sleep, because so many friends got the **happy** cause from tomorrow
- Today is Tuesday and I am to announce another project in Hiddu after ten rounds**! It's simple**

Prompt: "Today is Tuesday and I am"
Feature Vector: feature 2045 in residual stream pre of layer 7
Feature Property: sadness
Generations:

- Today is Tuesday and I am **sad** to announce that **someone was killed in a car crash** at C&C
- Today is Tuesday and I amistic, so you may have noticed something about its **very small body of work**
- Today is Tuesday and **I am to be buried**. This day's action, like any great soul searching on
- Today is Tuesday and I am as good a person **to die** by my family (except that I had to

Prompt: "C'est mardi et je suis"
Feature Vector: feature 2392 in residual stream pre of layer 7
Feature Property: happiness
Generations:

- C'est mardi et je suis with **la vie**, il est une **happyité**
- C'est mardi et je suis de la fait de **l'esprit des vivants**
- C'est mardi et je suis **couste** il peut in effantment qui **nouveau pass**

Note that I use a feature that fires for “happy” to steer the model’s behavior in French. This is not quite elegant but it turns out that it works to some extent! Though there are mistakes in spelling and grammar, the semantic tends to be quite positive! This means that the feature firing for English has an **indirect** influence on the model’s behavior in French.

2 Training an SAE on Phi-2

This is not what I mainly work at. Actually I did the experiments in section 1 during the training process. I treat this part just as a try, and the result was not so satisfactory to me.

When I started to train an SAE, I chose ChatGLM3-6B because I worked with the GLM team before and I'm familiar with the model. When I went through the Transformerlens doc, I found that the GLM models were not in the list. I guess it's because of the special structure or some special techniques of GLMs that make it harder to add them into Transformerlens, or simply nobody tried that before. I hope I can figure it out later because GLM models are really excellent bilingual language models.

Later I used Qwen2-7B which is also a bilingual model. Training SAEs on big models (for example, 7B) on limited computing resources is really a painful experience because most of the time I am struggling with cuda memories! Since I use the framework of SAELens and parallel training hasn't

been deployed well enough, I have to coordinate with limited GPUs. I plan to develop a parallel training framework and better memory management for SAELens later (if I have time and energy).

I trained an SAE on the MLP out of layer 4 in Qwen2-7B. The MSE loss was quite strange because it moved up and down around a high level and had no tendency of dropping. Later I changed the model to Phi2 and tried again on MLP out, but the same thing happened.

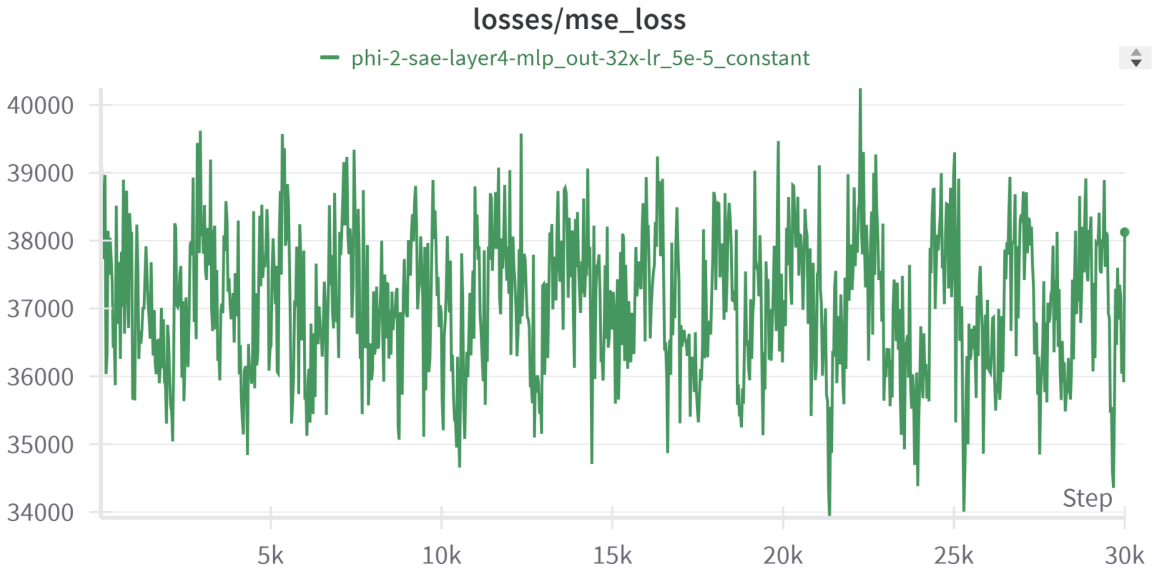

Fig 11: Training curve for MSE loss when training an SAE on the MLP out of Phi2

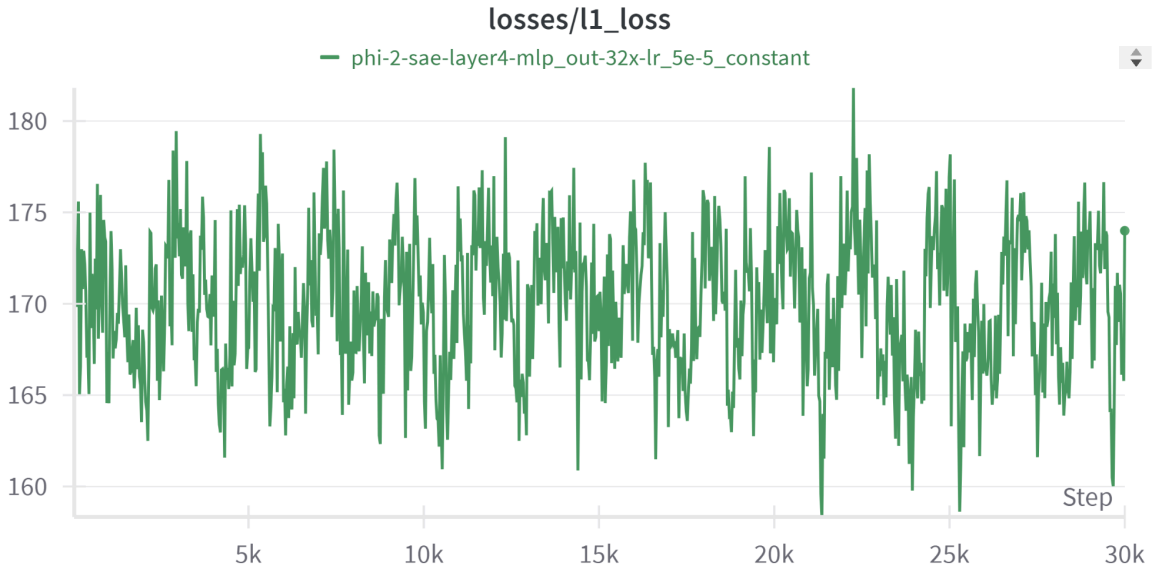

Fig 12: Training curve for L1 loss when training an SAE on the MLP out of Phi2

Later I changed the position of the SAE to residual stream, and the loss curve changed a little but was still confusing and showed the same pattern as before.

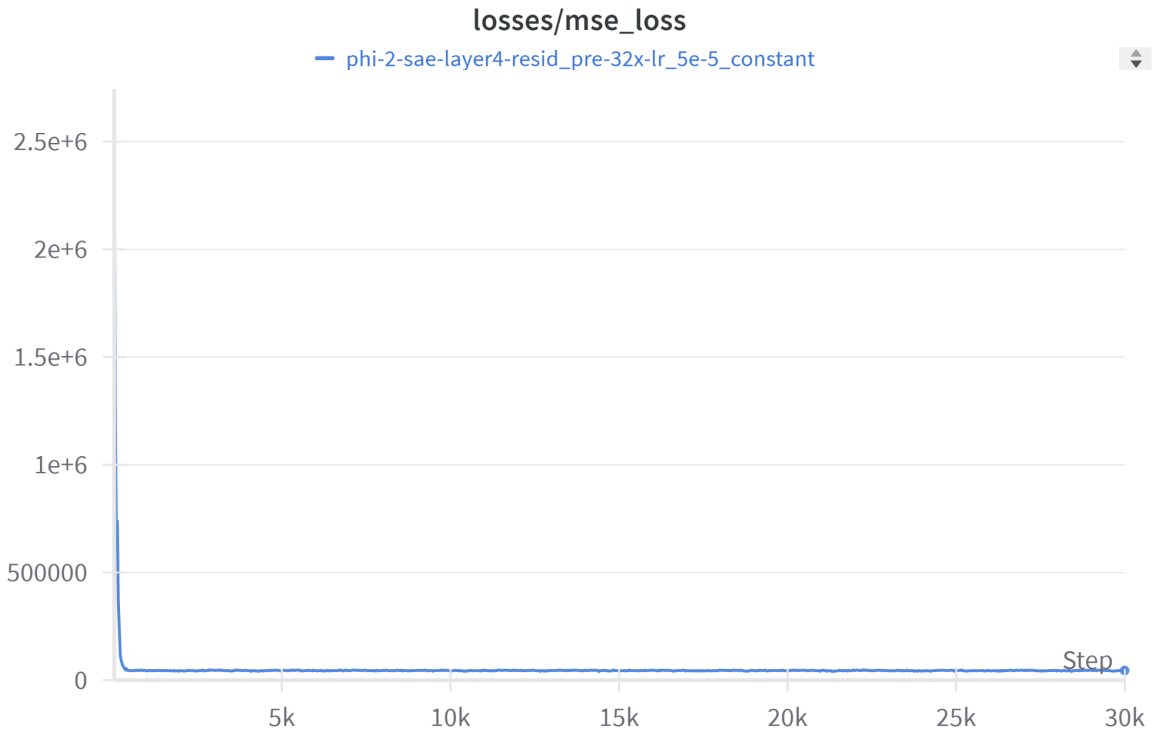

Fig 13: Training curve for MSE loss when training an SAE on the residual pre stream out of Phi2

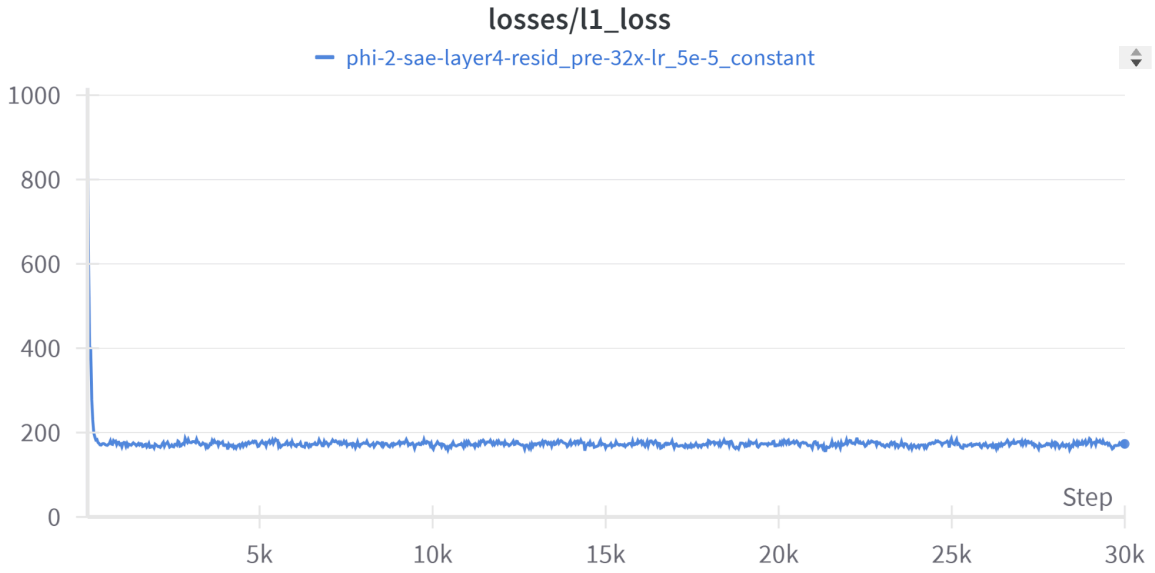

Fig 14: Training curve for L1 loss when training an SAE on the residual pre stream out of Phi2

I doubt that it's due to my fault in data processing? I downloaded the WanJuan dataset which was once used for training InternLM. I tokenized it using the tokenizer of each model. I haven't found any problem with that. The most confusing thing is that when I ran the example provided by SAELens (trained on the TinyStory model), I found that the loss increased! I think I have to delve deep into the training framework later.

3 Some bugs I found

I found some bugs in Arena 3.0 and SAELens. For example, the function plot_features_in_2d in Arena 3.0 failed when I ran it. I have fixed it and sent a pull request to the Arena team. For other bugs I will re-check them carefully and contribute to the public repository.

Thanks for reading !!!