
SUPPLEMENTARY MATERIAL

SnakeChat: a conversational-AI based app for snake classification

Jorge Guerra Pires ,¹

¹Founder at IdeaCoding Lab / JovemPesquisador.com, Brazil

*jorgeguerrapires@yahoo.com.br

Abstract

This is the extended discussion section for "SnakeChat: a conversational-AI based app for snake classification".

Keywords: biology; bioinformatics; conversational artificial intelligence; Snakes; openAI; chatGPT; chatbots.

1 Extended Results and Discussion

At the end of our last paper [Pires and Dias Braga \(2023\)](#), we proposed an app that would connect SnakeFace with the textual capabilities of chatGPT (Fig.1).

SnakeFace is a transfer learning based tool created to classify snakes. Different from general-purpose computer vision models such as MobileNet and [INaturalist Computer Vision](#), the model was designed to identify Brazilian species. General models, such as MobileNet, tend to overlook Brazilian species; they are good at saying whether we have a snake on the image, [even adding a label](#), but bad at actually finding the species. As we are going to see, also the vision model of openAI also overlook Brazilian species. I have also compared with other models (e.g., BARD from Google, see Supplementary Material). Our goal is making SnakeFace a part of a conversational AI, and still getting it right for native species from Brazil. See that BARD also can work as a general-purpose snake classification chatbot, see SM for a comparison.

If you see with attention, our first idealization of SnakeChat (Fig. 1) was by using chatGPT for guiding which algorithm should be used on a possible design with small models (models designed to identify snakes that have something in common, such as being false coral snake). We had in mind a manual function calling, now openAI made available an option to automatically do that. Furthermore, since we have already done a couple of tests, we knew chatGPT could get close to guessing snakes by textual descriptions. Then we thought: that could be useful. It is possible that on a real-accident with snakes, a textual description is the only information we may have for deciding on the serum.

The motivation on specialized models is that as we grow bigger, also, the accuracy per class will decline. When we have general-purpose computer vision models, such as

MobileNet, it is impossible to make sure a species is well-classified. In the case of INaturalist, making a mistake will just imply that someone else will correct you. On a real application, such as finding snake serum, a mistake may be costly. Hence, the higher the accuracy per class, the best.

One interesting observation about snakes, compared to other animals. They are geographically-focused species. If you consider a healthcare center which may want to use the system, it could be severed with a local model. Trained and adapted for them. Thus, increasing the accuracy, and minimizing misclassifications. Even though some species of snake may have a broad geographical distribution, when we fix a geographical radius of interest, the possible species to be found on that region will drop drastically. Therefore, using general models will just increase the misclassifications, without a real trade-off at stake.

Small models are interesting due to two reasons mainly: i) when you retrain the model, we do not lose what we did good at the training stage, with a big model, when we retrain, we lose what we did well on the previous training; ii) small models may avoid mistakes between snakes that are very different.

The model does not see as we see, it seems to see globally, the surrounding can deceive the model to classify badly; one proof of that is that if you remove the background, surprisingly, it does not improve the prediction, it may worsen the prediction. Also, one of the snakes we tested (see Supplementary Material) was identified as a boat by MobileNet, and as a duck by INaturalist Vision Model. It makes sense since those two misclassifications are generally seen in water, and the snake was in water. Also, a very easy snake to classify was misclassified for being on a human hand (see SM).

One big issue we face with big models: as we add new species, we loose what we did right. For instance, during

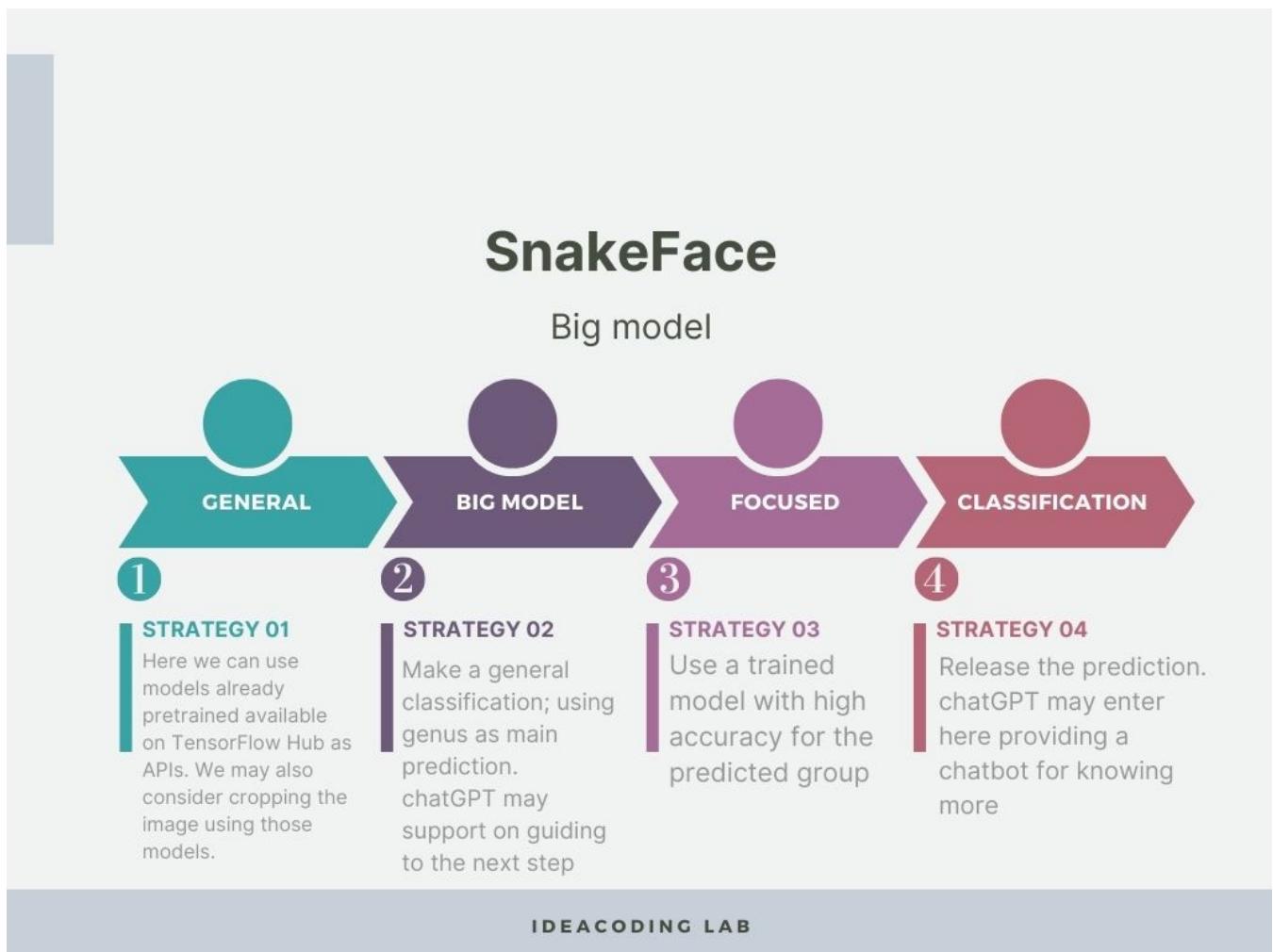


Figure 1: First idealization of snakeChat. Source: [Pires and Dias Braga \(2023\)](#)

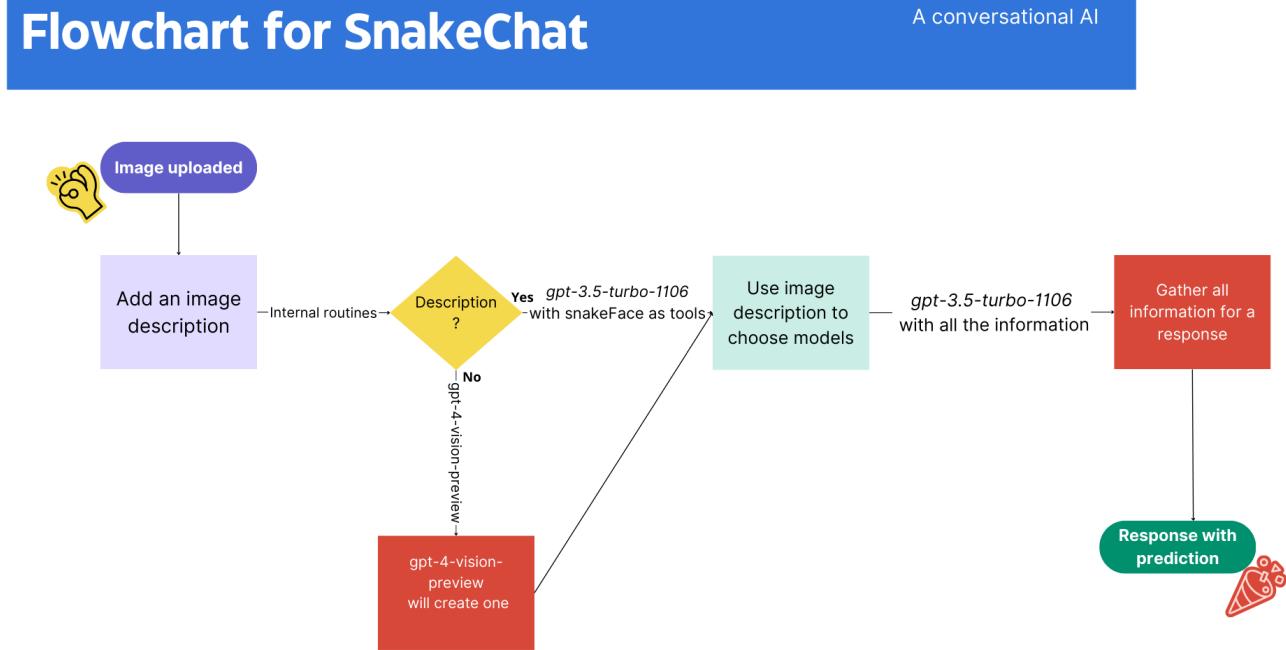


Figure 2: Executed version of SnakeChat based on SnakeFace vision. See that for the last test we present we have replaced *gpt-3.5-turbo-1106* by *gpt-4-1106-preview*. **Legend:** the user uploads an image. If the user does not provide an image description, we use openAI API for creating an image textual description. Once we have the image description, we use openAI API to decide which model(s) from SnakeFace to use for making a guess using computer vision. Then, finally, we gather all the information (SnakeFace + openAI API) for making a final guess.

the first versions from SnakeFace, we could classify Fig. 4. They are two problematic images: one is not clear, the other is a mimicking snake which does its work pretty well. The model during training is evaluated globally (a global error function is used to guide and finish the training process): it is not evaluated for a specific image. This is necessary since models are trained to operate on uncertainties, not for specific images.

Recently, in a single announcement, openAI, the group behind chatGPT, released new tools for their APIs. Those new releases are what we need to make SnakeChat possible. Before, we were just thinking about chatGPT as just support, and informative for the user. Now, we are actually using their API for making our app smarter. Thus, it was integrated into SnakeFace.

On the following image (Fig. 2), we have how we executed fig. Fig. 1 using as pivot the latest APIs from openAI. Different from the initial plan to use just chatGPT API, we could go further thanks to the latest releases from openAI as APIs.

In the upcoming sections, we are going to discuss more closely three of the results presented on Table I. The results on this table are for the first version of the chatbot, see SM for more variations of configurations.

See the supplementary material for more examples from the table.

It is important to highlight that this table does not

present a statistical analysis. For doing so, we would need a considerable amount of work. openAI APIs are constantly updated, and SnakeFace is a prototype, new training will create new emergent behaviors on SnakeChat. SnakeChat is built on top of those models, thus, their behavior will change the behavior of SnakeChat. Thus, at the current moment, we consider useless any statistical analysis, which will entail also financial costs since the openAI APIs are not free of charge. Also, the latest configurations include more options. It is hard to predict how a LLMs will behave for each human-text input. We would need to test all the possibilities, variations and more. We also expect to add more features. The best approach is each user play with different settings, and see what works best for them.

1.1 Conversations

See SM for more conversations, using different configurations.

1.1.1 *Oxyrhopus rhombifer*

We are going to start with what the current app does quite well. Rarely, it will misidentify (Table I).

First, we need to bring to attention, as so you can understand better what to expect and from where it may come some undesired behaviors (e.g., stochasticity).

Table I: General behavior for the chatbot.

Species	SnakeFace	openAI API	Final result
<i>Oxyrhopus rhombifer</i>	got it right when using model for both true and false coral snakes	coral snake (wrong)	<i>Oxyrhopus rhombifer</i> (correct)
<i>Erythrolamprus aesculapii</i>	got it wrong since openAPI API called the wrong model	coral snake (wrong)	<i>Micrurus frontalis</i> (wrong)
<i>Oxyrhopus rhombifer</i>	got it right when using both models (just fake corals and both)	coral snake (wrong)	<i>Oxyrhopus rhombifer</i> (correct)
<i>Bothrops Jararaca</i>	got it right, openAPI API called the right model.	caninana (wrong)	<i>Bothrops Jararaca</i> (correct)
<i>Erythrolamprus aesculapii</i>	got it wrong since openAPI API called the wrong model	coral snake (wrong)	<i>Micrurus corallinus</i> (wrong)
<i>Oxyrhopus rhombifer</i>	got it right with two models, but misclassified on one model	coral snake (wrong)	<i>Micrurus corallinus</i> (wrong)
<i>Bothrops Jararaca</i>	got it right since openAPI API called the right model.	Jararaca (correct)	<i>Bothrops Jararaca</i> (correct)

Important. this is not a statistical analysis, it is a general behavior demonstration.

Legend. The upper section is for *gpt-3.5-turbo-1106* as final model (fourth column) and also as tool selector. Lower section is for *gpt-4-1106-preview* as final model, also as model selector. The second section (middle) is for *gpt-4-1106-preview* just on the guessing stage, for selecting which model to use from SnakeFace. See SM for examples of using *gpt-4-1106-preview* all the way.

Note. The mistakes of SnakeFace (second column) is influenced when the wrong model is used, that is, it is influenced by openAI API (third column). The core idea of using openAI API was precisely to use the right model, and avoid the need of a big model. We believe that smaller models will give more control over the possible errors made by SnakeFace. openAI API can fix the error on a last call (third column), where it should gather all the information, and make a guess.

- The algorithms from openAIs we are using are stochastic: they offer different responses upon the same input. This means that sometimes the response from openAI API may not be helpful as we hope. It seems to improve with their GPT 4 model, see our last conversation;
- SnakeFace is deterministic: the same input will always lead to the same output. This means that it may get it right, but be second-guessed by openAI API. This may occasionally lead to a confusing final response;
- We have not yet figured it out a good solution, but you can read the probabilities, and pick the highest one, but for some reasons, openAI API will pick not necessarily the highest probability as final response. It can be confusing. It seems to trust more the textual description than the numbers. It seems to improve with their GPT 4 model, see our last conversation (see lower sections from [Table I](#)).

It is important to bring to attention: variations on the responses are due to openAI APIs.

We have observed two common behaviors: i) when openAI API helps on the final response; ii) situations where it does not help, creating confusion.

One example is when SnakeFace gets it right, but openAI API guesses it wrong. Since we are logging all the responses, someone with the ability to handle those contractions may still gain from the app. Nonetheless, our hope would be an app that handles a single, simple, and correct response. It is not our current case. We

saw apparent improvements when using their GPT 4 model: it does not call unnecessary models, therefore, less unnecessary numbers to confuse the final response. However, as you can see from [Table I](#), this behavior from GPT 4 API is not deterministic, it is not always the same. The lower sections all use GPT 4 API, as so we can have an idea on how it may be used to enhance the SnakeChat behavior.

Another example, see SM, is when a human-entered snake description was provided. openAI API called the wrong model. By calling the wrong model, SnakeFace provided a wrong prediction, which lead to a wrong final response. What is interesting: GPT 4 API created an entire argumentation to back up the wrong prediction, which was inconsistent with the snake description. It is a sort of "hallucination". I have entered the same conclusion on Bing Chat and Bard, they are both LLMs, asking for checking for inconsistency, they both failed to see that the snake description was inconsistent with the predicted snake. It is most likely that anyone minimally knowledgeable would see the inconstancy.

One nice feature from openAI API that may come handy in the future is that they allow to train their textual models, called [fine-tuning](#); and then the model will be available to be called. It is possible in the future try to improve those textual guesses by training their model, and using the fine-tuned model instead of the generic one we are using. It is similar to what we did with SnakeFace: we fine-tuned a generic model called MobileNet, that could not identify precisely Brazilian snakes. MobileNet was



Figure 3: *Oxyrhopus rhombifer*. *Oxyrhopus rhombifer*, the Amazon false coral snake, is a species of snake in the family Colubridae. The species is native to Brazil, Bolivia, Paraguay, Uruguay, and Argentina. Source: Wikipedia.

trained for generic objects. It is safe to assume that in some situations, a text-description of the snake is all someone can provide after an accident with the snake, which most likely will run away. Or a human-memory description.

Our input image is below (Fig. 3): it is a clear image, no visual distractions, simple background. A good question is how the model handles bad images (e.g., ??): it is uncertain. It may see, or may not see. As SM, there a case where it is hard even for humans to see the snake hidden on the vegetation, but the model saw, even though it did a misclassification with a similar snake. Other models, such as MobileNet, failed to see the snake at all. The ideal is always a clear image, with no obstructions.

Keep in mind: it is a matter of training, since the image mentioned was classified on previous training, but we lost it as we added new species. It is something we cannot control.

We are going to test first a human-generated image description. As SM, we have tested in PT since this is the final goal, and we want to make sure the chatbot can handle Portuguese properly.

The snake is black, red, and white. The colors are in strips. The strips black are bigger, whereas the red ones are smaller.

We asked *gpt-3.5-turbo-1106* guessing the snake from the description. It is a parameter for each function; at the current state, it is a dummy parameter, we are just using as chat logs. It guessed as coral snake, which is wrong. As said before, the behavior of openAI APIs is stochastic, in some tests, they actually guessed it right.

Response when calling model for both true and false coral snakes.

Probabilidade de 82% de ser uma *Oxyrhopus rhombifer* usando SnakeFace, um sistema de identificação por imagens. [correct]

Obs. the system was designed to respond in Portuguese, thus, you will see hybrid responses.

Second call, using the generic model:

Probabilidade de 58% de ser uma *Micrurus corallinus* [this is a true coral snake] usando SnakeFace, um sistema de identificação por imagens.

Remember that openAI API decides which model(s) to call: those calls were done automatically by openAI API. It has at its disposal three models, discussed in Pires and Dias Braga (2023); see the method section for minimal details therein. See SM where we have tested providing more models, and more configurations.

If we see logs, *gpt-3.5-turbo-1106* decided to call two models we made available from SnakeFace: *true_and_false_corals* and *generic_model_snake*; the former is used to classify false and true coral snakes, it was created to try to separate them from each other, see Pires and Dias Braga (2023); whereas the latter is for generic snakes, it was meant for a possible big model, generic. It did not call for the false coral snake model: which makes sense since its guess was wrong, it was coral snake. See that in other tests we have done, it actually guessed it was a false coral snake; remember that its behavior is stochastic.

Thus, the first call to SnakeFace rejected *gpt-3.5-turbo-1106* guess on the snake, whereas the second guess confirmed.

However, on this case it actually joined properly the contradicting information:

The snake in the photo is most likely an *Oxyrhopus rhombifer*, also known as the Mehuaco or South American False Coral Snake. It is often confused with the *Micrurus corallinus*, or the Coral snake, which has similar coloration.

This response was perfect: *gpt-3.5-turbo-1106* was able to actually handle the fact that the generic model and *gpt-3.5-turbo-1106* confused the snakes that are alike. The generic model of SnakeFace gave 15% probability for *Oxyrhopus rhombifer*. The false vs. true coral snake gave lower than 1% of being a *Micrurus corallinus*, which is good. It reinforces our call for specialized small models for SnakeFace instead of big models, as we are proposing herein.

If we make an analogy to human: *gpt-3.5-turbo-1106* on this case specifically did not fall into confirmation bias.

It seems based on observations during the last conversation using *gpt-4-1106-preview*: their GPT4 model seems to be less likely to fall into confirmation bias, as we use it here (the model uses SnakeFace mistakes to confirm their initial guess).

One may ask ourselves how other researches are doing. We did a small literature survey on our other paper Pires and Dias Braga (2023), we are not going to replicate it herein. In essence, there are several researchers also applying transfer learning to snake classification, it seems a topic that got researchers interested, and the results are in the same direction as we are going. Except, none of them is using chatbots, or even, JavaScript. Those two details may give us an advantage on building an app. Also, they seem to focus on big models. Big models can increase problems on accuracy per class (it may perform well on a set of classes, but badly on others). Low accuracy per class can become an issue when it is important to avoid misclassifications (e.g., finding a snake serum).

One issue we found is that those models are built using different models and they are in python, we are coding in

JavaScript. None of those models made available, as far as we know, APIs available, that we could just plug in to our SnakeChat and test. We are opened for collaborations with other researchers to test their models inside SnakeChat. All the models used herein were used as legos: we can just remove one and put others for testing (see . Fig. 1). This approach makes it easier to compare models, as long as they are easily available for being used.

Now, we are going to test the system without human-image description: the AI (*gpt-4-vision-preview*) will make the description.

The snake in the image has a patterned coloration with three primary colors: black, red, and white. The black color is prominent and appears in thick bands or stripes that run the length of the snake's body. Between the black bands are patches of red color, which are themselves outlined with a thinner white or cream border that accentuates the red areas. This tricolor pattern is highly striking and gives the snake a distinct and bold appearance. The arrangement of colors suggests a repeating and systematic pattern, which is often characteristic of certain snake species, particularly those that use their distinct coloration as a warning or as camouflage.

The snake guess was still coral, even with its own image description. We could test fine-tuning their algorithm, something we may test in the future. For this approach, we need image description and their respective snakes. We are not even sure if that would be possible.

In that case, the final response was wrong, even though two of SnakeFace models predicted the correct snake (*Oxyrhopus rhombifer*) : 82% and 69%; one of the models misclassified as *Micrurus corallinus* (58%)

But still, the final response was:

The snake in the photo appears to be a coral snake, specifically a *Micrurus corallinus*. The tricolor pattern and the prominence of black, red, and white colors are characteristic of coral snakes.

As we said before, someone reading all the logs may say: "it is two against one", and pick the correct guess. Making an analogy to human psychology: here their algorithm felt into confirmation bias.

We tried out their latest model, GPT 4 (*gpt-4-1106-preview*), and also got similar response:

Based on the description provided, the snake appears to resemble the coloration and pattern commonly seen in a Coral Snake or a species mimicking a Coral Snake, such as a Milk Snake or a King Snake. [that was a better guess!]

Coral Snakes, which belong to the genus *Micrurus* or *Micruroides* in the New World, and the *Calliophis*, *Hemibungarus*, or *Sinomicrurus* genera in the Old World, are known for their distinctive red, black, and yellow/white banding. However, the phrase "*red touch black, venom lack; red touch yellow, kill a fellow*" is a mnemonic often used to distinguish the venomous Coral Snake from its non-venomous mimics in some regions (like the Milk Snake or the King Snake) where the order of the colors varies. Nonetheless, this rhyme is not reliable for identifying Coral Snakes globally, as color patterns can differ by region.

Given that the description mentions red and white (or cream), and considering that some Coral Snake species may have a yellowish or whitish band instead of a clear yellow, the snake could potentially be a Coral Snake. On the other hand, since *the description does not specify yellow but white or cream, it might be more likely the snake in question*

What is interesting about this response, the algorithm gave the answer when it has written a common saying amongst people: "*red touch black, venom lack; red touch*

yellow, kill a fellow". There is no yellow on the snake coloration. It finished: *the description does not specify yellow but white or cream, it might be more likely the snake in question*. The conclusion was strange, it seemed it would actually conclude right.

But when we asked to make it short: Coral Snake. Same as the current version we are using. Thus, what remains is trying to fine tuning their model. We are going to leave that experiment for a possible future experiment.

1.2 *Erythrolamprus aesculapii*

As we brought to attention in our previous work [Pires and Dias Braga \(2023\)](#): this snake is tough to classify since it was able to fool our models most of the time, mimicking a coral snake. One thing we said in our previous paper: our goal should be avoiding identifying venomous snake as no-venomous. We called it error type II. In fact, it was the main focus of another research [Felipe Guimarães Dos Santos \(2021\)](#). Since we are using SnakeFace, we did not expect different! See that we were able during the training to classify it properly, this image. The problem is that the model is evaluated as good and bad globally, not for a specific image: when you see a good result from the image, it is an overall model behaviour. The version we have used now was retrained after the paper, and it fooled the current model.

We added the description manually for this case:

The snake is red, white and black. The colors are in strips. The red strip is the biggest ones, whereas the black and white ones seems at the same size.

We asked also their latest model to guess, it guess equally wrong compared to the current version we are using from their API. Guessed coral snake (see that this behavior is persistant for coral snakes, see [Table I](#)).

As a result, it decided to use just one model. Guessed as *Micrurus frontalis* (100%); indeed, compared . Fig. 4 and Fig. 5, it is hard to make the difference just using images. We believe that if the user provides details that are not in the image, we believe the whole system (i.e., SnakeChat) can get it right. The issue with that is that just an expert would think to provide information about the head, tail and more. We have in mind a system for non-experts. We have experimented with asking GPT 4 Vision to focus on this details when generating an image description, and the result is interesting (see SM). When those details are visible in the image, it can describe those parts of the snake, even for images as bad as ??.

1.3 *Bothrops Jararaca*

We are going to try something different; we tested as we are, and the guesses were bad, calling *Bothrops jararaca* a coral snake. It may lead to right answer corrected by SnakeFace and accepted, but also to what we have already seen: a sort of confirmation bias. See lower sections from [Table I](#).

So far, we used their cheapest model, let's try out GPT 4 (*gpt-4-1106-preview*). This is their latest model, and widely commented upon, with intriguing results like



Figure 4: *Erythrolamprus aesculapii*. Source: Pires and Dias Braga (2023)



Figure 5: *Micrurus frontalis*; compare with fig. 4, they are essentially the same visually. Source: Wikipedia



Figure 6: *Bothrops jararaca*. Source: Wikipedia.

reading images and graphs with high level of intelligence.

Description using the same algorithm we have already used so far:

The snake in the image has a patterned appearance with a mix of neutral and earthy tones. Its scales seem to have a sequence of dark brown or black, beige, and lighter brown hues. The patterns are irregular, with some semblance of bands or blotches that alternate along the length of its body. The darker bands are bolder and more pronounced, providing a type of camouflage among the twigs and leaves on the ground. There do not appear to be any red or white stripes as part of this snake's coloration.

Surprise as it may seem, this description left to guessing as coral snake with the model used so far. We are changing both the function calling algorithm and the final response algorithm (see 1; see also Table I, lower section). Thus, just the image description algorithm was kept; at the moment, it is the only option they provide to describe images.

Their guess was jararaca. Much better. We have done the upload three times, with the same result, We found that the *gpt-4-1106-preview* is more predictable than *gpt-3.5-turbo-1106*. In their website, they suggest to fine-tune *gpt-3.5-turbo-1106* for getting this better result, just using the outputs of *gpt-4-1106-preview*. *gpt-4-1106-preview* is more expensive to use. Regarding the final cost of the fine-tuned model, it does not seem a good deal once the fine-tuned model will be more expensive than using their GPT 4 already trained. It is something we must consider if needed.

Our model confirmed the textual guess *Bothrops jararaca* (97%).

What is interesting, it also avoided calling unnecessary functions. When guessing as coral snake, as did the previous setting-up, it will call unnecessary functions, creating noise in the final response: creating space for wrong guessing the snake. On the calls we did with this image, they always called just one model: the general model; different from the previous set up, that seems to be calling unnecessary functions, and creating unnecessary results to interpreter.

Final response:

Based on the description provided, the snake in the photo is likely to be

a Bothrops jararaca, commonly known as the jararaca. It is a species of pit viper found in South America, and its appearance, characterized by a patterned skin with a sequence of dark brown or black, beige, and lighter brown hues, matches the description given. This type of coloration allows the snake to blend into the forest floor, making it well-camouflaged among leaves and twigs. Please note that without the actual photo, this identification should be considered a hypothesis and should be verified with a visual confirmation or by a herpetologist.

Best of all, this result tends to repeat itself if you upload the same image. It is good! See [Table 1](#) for a case where it guessed wrong, but still we got it right.

1.4 How SnakeChat could be adapted to similar applications

Even though we have focused on snakes, this approach can be applied to any case using images. The final result will be the same: a chatbot with state-of-the-art capabilities using images, a *conversational AI*. This includes, as we stressed on our other paper [Pires and Dias Braga \(2023\)](#), other species beyond snakes. As long as you can separate the species from the images, you can build a similar system for your desired application.

For instance, we have done some tests and discussions with a medical doctor researcher working on those applications as well. You can either train what they call GPTs (which is their way to make their models even more accessible to no programmers), or use the same path we used here: Teachable Machine, and use the models as we did, as external APIs. We have tested Teachable Machine applied to medical images, and the results are promising. In fact, our next paper may be on a system we are working on, but for medical images instead of Snakes. Thanks to how the openAI APIs are built, the transition between applications is straightforward. We have done a first prototype here [Pires \(2023b\)](#).

Our path has the advantage that Teachable Machine is free of charge to use, all the way, including hosting the model on Google. This means you do not have to pay for the vision part. Also, the GPTs are not available as APIs, when we wrote this paper. The computer vision model we are using (namely, *gpt-4-vision-preview*) does not allow fine-tuning, [as they pinpoint on their own documentation](#). [We have also here a comment on their forum](#): it may be released in the upcoming feature as an API.

Another option is that you have a model from your group, which was well-trained for a specific task. Even though the models from openAI are advanced, they seem to perform poorly on specialized problems. In a couple of tests done by another researcher, and confirmed by us in our own scenarios, if you ask their computer vision models to make diagnoses from images, it will either make a wrong diagnosis, or refuse to do so [Pires \(2023b\)](#). But they are very good at describing images, as we did. This description can be used to choose more specialized models, as we did.

There are plenty of literature studying the openAI models applied to diagnosis in medicine, their limitations and strengths, as example, but not exhaustive. [Ghosn et al. \(2023\)](#) compared the GPT 3.5 and GPT 4.0. As we expected, GPT 4 was better at answering questions from an standard examination in radiology, and also explaining the answers.

1.5 Challenges

This section was created based on a public review we received on [Qeios.com](#), on the first draft of this paper. We have found very insightful the challenges the reviewer raised, and found it interesting to add those challenges and their respective discussions on this final manuscript. Hopefully, it will support others on understanding our challenges.

1.5.1 Potential for disinformation

There is a risk for disinformation on those LLMs. One example is on the SM, where the LLM created an entire argumentation around a wrong prediction. That is a topic that we may need to handle on the future. Several solutions may tested. As example, it is possible to provide texts for the models, and ask the model to base their answer on this material. That would require a dataset of texts written by experts, which may be a long-term task. Other option is experimenting with fine-tuning their models. Those are all possible ways to handle disinformation on those chatbots.

1.5.2 Hallucination

Hallucination is largely associated to LLMs, it can be also seen on computer vision models. For instance, MobileNet saw a boat where were a snake, or even a crocodile. See SM. This is something we must consider solutions to mitigate. One way is to avoid using the wrong model, even though we have tested some possibilities, such as using MobileNet for having a general classifications, I am planning to also experiment with other alternatives (e.g., INaturalist).

1.5.3 Integration complexity/dependency on External APIs

Integrating external data sources (APIs), like INaturalist and openAI APIs, as we did and are planning to do, adds complexity to the system. Relying on external APIs, such as OpenAI and INaturalist or any other we may add in the future, introduces a dependency on the availability and stability of these services. Any disruptions or changes in these APIs could impact the functionality of the system. That is indeed a possibility when our system is built using external parts, third-party software/APIs/datasets. It is a trade-off: speed at building comes at a cost, lack of control of the system behavior. Moreover, it is hard to imagine we could build a dataset so rich as INaturalist, where people are posting their observations constantly, they even have a pretrained model on TensorFlow Hub [Pires and Dias Braga \(2023\)](#). In our view, this sharing-culture we built in the last decades is the best thing that ever happened to applied sciences, to collaborative endeavours as ours.

In fact, that was the detail that made it senseless a validation, as we did with SnakeFace [Pires and Dias Braga \(2023\)](#): we plotted errors graphs, calculated confusion matrices and more; we did a standard validation of neural network models, following well-accepted metrics and procedures. A single change on any of those external sources (i.e., APIs) will change the validity of the validation.

Systems like ours nowadays fall into the complex system category, a theory largely used on studying biological systems. Complex is not about size, when

seen from the complex system theory, it is about how hard it is to predict behaviors, as we like to say "the whole is greater than the sum of its parts", first coined by the philosopher Aristotle. In fact, we have used a term from complex systems theory on this paper, we have used the term 'emergent property' for designating SnakeChat behavior, it is the summation of SnakeFace and OpenAI APIs working synergically, which by themselves depends on other parts that we cannot see and predict; SnakeFace is more predictable than openAI APIs for being a deterministic model.

One big criticism on openAI is the fact that their algorithms are closed, no one really knows what is inside the magic hat. They keep a page, and an e-mail list, where they constantly post instabilities, and what they are doing to handle the issues. It is constantly active: it seems they are always making changes in their models, which may create undesirable behaviors. For instance, when we simulated the results on [Table I](#), we had the impression that GPT 4 API was more predictable, but now it seems it is behaving as chatGPT API, just an impression.

A term that is gaining attention is [software entropy](#). This term is used on software testing [Khorikov \(2020\)](#). We are using Angular as framework for SnakeChat, Angular was created to be tested [Jesse Palmer \(2018\)](#). Software testing is done precisely to handle, or at least, mitigate software undesirable behaviors, making them more predictable. External APIs and datasets is a big challenge. We do what is called *mocks*: we just ignore them. We create a local version of APIs and datasets, which behaviors in a predictable way, which is unrealistic. The idea is that each groups should be responsible for their quality control. Therefore, generally, we do not try to control external APIs/datasets. The best we can do is choosing reliable sources. At the current stage, we are not concerned about software testing, we are prototyping. There are whole organizations just focused on software testings.

As we can see from [Table I](#), the behavior of openAI APIs is indeed stochastic, and it seems they are working to increase predictability of their APIs, which for sure will affect ours.

When we do a scientific search for terms like "chatbots in biology", most of the results, when we find any, are medicine driven, and they are using precisely openAI solutions. It is impossible to imagine someone building a whole large language model, just to classify snakes. The technology behind openAI was not created by them, it was around and it was created by scholars and Google researchers. Therefore, the fact that no scientific group created such a LLM is a sign that it is unfeasible considering creating our own LLM. The costs of openAI is very low, based on what it can do. Thus, it is hard to imagine other groups would build such a system with such a low cost. Of course, we do have open sources groups working on that, and their results are interesting. Hopefully, they will also create LLMs, like openAI APIs: easy to use, with almost zero setting ups.

1.5.4 User Understanding and Expectations

This is an interesting point, and we have discussed it previously in [Pires \(2022\)](#). We believe, similar to the mindset from openAI: we learn by interacting with the

user. That is why we have launched all those prototypes. It is hard to handle user's expectation when you are using anything related to chatGPT, and we have seen that firsthand: all the attention around chatGPT can backfire, people seem to expect much more than chatGPT-related systems can delivery. Understanding the user may take time, but we see a general user, without expertise in biology. We also see healthcare centers, and teaching institutions as final users.

1.5.5 Scalability Challenges

We are not at this stage, but certainly scalability is something to consider, it is largely studied on start-ups contexts [Blank and Dorf \(2012\)](#). One challenge of scalability is adding new species, there are more than 2.000 worldwide, and the best model we know was able to classify about 700 species; we discuss that on our other paper [Pires and Dias Braga \(2023\)](#). One solution is on this paper: work with small models, and use openAI API to handle them intelligently. When we add new species, we cannot know what will happen with the current results. For instance, on early versions of the general model from SnakeFace, we could actually classify both ?? and [Fig. 4](#), two hard images to classify. As we added new species, we lost this impressive result.

Another challenge, which is discussed in [Pires and Dias Braga \(2023\)](#), is annotation, a good dataset. For training, we need images and their respective tags. Which have partially been solved using Google Image. Nonetheless, that is something to pay attention to. Our latest models were built using INaturalist, which is much better. They have a rich community, and annotations are under consensus. Also, their images are real, they are taken by users on real-scenarios. This is interesting for training the model.

One details in our favor: big models tends to lose accuracy per class. We can either build chatbots for regions (e.g., healthcare centers) or change the models according to the user geographical position. Also, we are considering a model that could better be used to chose small models. Small models could make the scalability easier.

1.5.6 Pricing

It is always nice to build such a app for free, or at reduced cost. Nonetheless, there are costs involved that will have to be taken into account on a production model. [OpenAI API has a pricing](#) that changes according to the model used. Their pricing is "pay-as-you-go": they charge as you use their models, not a monthly/annual billing. It is hard to make an estimation now since the pricing will change on which model is used. For instance, the GPT 4 Vision charges high resolution image higher. Since we are using more than one model at once, depending on the configuration (see SM), the pricing will vary according to the configuration chosen by the user.

One solution for creating a free version is replacing the function calling by MobileNet, which I have shown not to do a very good job with Brazilian snakes. it will work, but with less accuracy. I have also showed that it can be used instead of the image description model from openAI. Again, the result may not be good as it is now, see SM.

There are several open source LLMs gaining

momentum as the LLMs gain attention for their advanced capabilities, and they may become in the future an option. I need to get more familiar with them, and see if it is possible to replace the openAI LLMs with them, whenever possible for reducing costs with LLMs.

There is also the server cost, I am using Heroku. Their pricing is for the entire server, it is possible to deploy several apps on the same plan. For this, options are Wix Site, which allows to code in JavaScript, and charge for premium options only. Another option is GitHub Pages. Of course, all those options come with limitations. More options may certainly appear as we go.

1.5.7 Limited Control over Training Data

It is true that we have limited control over what openAI API says. One solution is using a text-dataset on snakes we can classify, created by experts. It can be just a blog, since we can access blog posts and send the content to openAI API to use as reference to their responses. The issue with this strategy is precisely having this expert-curated text dataset. Then, instead of using the knowledge from openAI API models, we just use it as a knowledge-mining tool. The latest APIs from openAI now can read almost 300 pages at once, they increased considerably their "attention window", a point of large criticism, and also launched Assistants, this is an way to allow them to actually manage attention-windows automatically. This is something we have been exploring largely on other applications where we use chatbots. See a discussion we did on data science Pires (2023a).

1.5.8 Adding more species

According to *Encyclopaedia Britannica*, there are nearly 3,000 species of snakes distributed nearly worldwide. Brazil is home to a diverse range of snake species, with nearly 400 species found in the country according to *Atlas of Brazilian Snakes*. The good question is whether we could scale up. The scaling up is not a issue of SnakeChat, which is a chatbot, it is an issue of SnakeFace Pires and Dias Braga (2023), the snake classifier under the hood when the chatbot tries to guess a snake.

Machine intelligence is different from humans': it does not change very much as you add more classes. Humans have a "normal intelligence" (a reference to the normal distribution), it is stronger around expertise, where machines have a "flatter intelligence", they tend to be good at a big number of classes at once. It is true that human experts can be better than machine learning in some scenarios, nonetheless, machine learning models once trained properly can be good at several cases at once. For classifying several species, one would need several experts. The expert will be good at their expertise, but average on the remaining. Machine learning once trained properly can be similar to several experts at once.

Even though Pires and Dias Braga (2023) explored a small number of species, it was explored strategically: similar snakes and diverse snakes. Those are like extremes. The current paper actually says that we do not need a huge model, but a number of small specialized models, and allow the chatbot to pick the proper model.

Thus, adding more species is more an issue of creating more models. Of course, this is time and resource -

consuming, and challenges may appear on the way. See Pires and Dias Braga (2023) for more discussions. Scaling this prototype to cover a broader range of snake species will be challenging and will require significantly more data and resources. Nonetheless, this is not an impossible task, we have shown it is possible.

1.5.9 GPT-4 API outperformed chatGPT API

Recent literature has highlighted the apparent superiority of GPT-4 over its predecessor GPT 3.5, particularly in the context of chat-based language models. Lubiana et al. (2023) provides tips for optimizing workflows using ChatGPT, which is powered by GPT-3.5 and GPT-4. Rahaman et al. (2023) compares the performance of GPT-4 with previous versions, showing improvements in training data, computation speed, answer accuracy, and overall performance. Rosoł et al. (2023) reveals that GPT-4 outperforms GPT-3.5 in a medical examination, with higher accuracy. These findings highlight the improved performance and capabilities of GPT-4 compared to GPT-3.5, supporting the claim of its superiority in the literature Lubiana et al. (2023); Rahaman et al. (2023); Rosoł et al. (2023).

We have mentioned regarding an apparent superiority of GPT-4 over chatGPT (i.e., GPT 3.5) for our case. It has not been done a careful statistical analysis, therefore, it is not safe for us making any bold claim on that direction. What we can say: the first simulations showed that GPT 4 tended to call just the needed model, which is a good behavior; but on the final simulations, which was done days later, they seemed to be equally in term of function calling. Calling wrong models as we have seen creates noise, unnecessary information, which can induces the final stage of the chatbot to arrive to wrong conclusion about the snake species. We also cannot say for sure why, nonetheless, we have noticed on another study we have done, which may be published regarding Section 1.4, is that the GPT 4 API is more "obedient", its higher cognitive ability makes it better at deciding which function to call. Thus, the function calling capability may work better with GPT 4 API. In fact, it was found that GPT 4 has a higher cognitive capability compared to GPT 3.5 West (2023).

1.5.10 Application on healthcare centers

One application I am considering is for the local healthcare center from Antônio Pereira (Ouro Preto). Snakes are local animals. There are groups of studies on the local university (Federal University of Ouro Preto) to register local snakes; SnakeFace Pires and Dias Braga (2023) was created in a collaboration with a student from this group. Since accident with snakes are real, the high-precision for a small number of snakes is good. Unless someone is by purpose entering a foreign snake, the number of snakes on a geographical location is small. SnakeFace/SnakeChat is good at a small number of snakes, different from general models which will misclassify those snakes (see TM). The type of antivenom serum required for treatment depends on the species of snake that caused the bite, and timing may be crucial.

1.5.11 Limitations and future works

In addition to all the testing I have done as part of the development process, I have also provided a set of conversation samples as Supplementary Material. SnakeChat/SnakeFace is not perfect, and it has several flaws that we hope to improve with time.

For instance, as I was finishing the writing of this manuscript version, I have found a snake on the same geographical region the species I have added was, during my walks. I knew the species was not on the dataset. After entering the image, the system have chosen a snake that had similar color, but a very different species. This happens because by prompt engineering on the openAI API, I have directly instructed to use the models even when the snake identified by MobileNet/GPT 4 Vision was different. This was done since initially the function calling would not call when the snake on the image was different from the ones on the models. As an example, MobileNet will identify *bothrops alternatus* as a rattlesnake, which are very different snakes. The function calling API would refuse to use any model since they are very different snakes.

Thus, we have a tradeoff, which seems without solution so far: a model was created since the general models we found currently online cannot see the Brazilian snakes well, especially, from the geographical region I have concentrated. However, when a wrong snake is entered, the model would still classify running the risk of making a totally wrong classification. I have actually tried BARD for the snake mentioned, and it did an even worse prediction. The solution was a manual combination of INaturalist with Google Image, which I have done manually, and took me minutes to merge the pieces of information. Therefore, our system can see what it was trained to see on where it is, but it can see what it was trained to see what it does not exist (i.e., a snake on wrong models).

This is a general issue on computer vision models, not specific for our case. If you enter an image with just grass and no snake, I have done that as a placebo, see SM, MobileNet saw several animals, including a rabbit. For a bad image, see SM, it saw even a crocodile. In our case, since the system is specialized, it can become easier to see those "hallucinations". I am considering solutions, such as using other general models rather than MobileNet, or GPT Vision. Nonetheless, it is important to keep in mind that some of the issues are bigger than our system, it is a computer vision issue, a general issue. There is no artificial general intelligence currently, all AI nowadays are designed to solve specific problems, as I have done. Some are more specialized (e.g., SnakeFace), other are more generic (e.g., BARD).

2 Conclusion

On this article, we have discussed a prototype for a conversational artificial intelligence (CAI) focused on snakes classifications. We have used the APIs from openAI as leverage, as so we did not have to build a CAI, just use it as leverage. Our model is an extension of SnakeFace, a transfer learning based model built to identify snakes. SnakeChat is focused on proving a more

human-like prediction: instead of just providing a species and probability, we provide as a textual response. It is a task-oriented CAI, but thanks to how openAI APIs are designed, shifting to a hybrid CAI is not that hard, also becoming a chat-oriented CAI.

In addition, we have used textual descriptions, that can be entered by the user or is generated by an artificial intelligence. Our simulations show a good integration between SnakeFace and OpenAI APIs, what we have named SnakeChat. Our next step would be: i) adding more species of snakes; ii) adding more information as so to enhance the predictions; iii) keep testing our model and keeping finding new algorithms that can make our system unique.

One way to add new information to the prediction is using INaturalist: they provide an API that can give back information about snakes, such as if they were ever seen on a given location. Several pieces of information, such as textual descriptions, probabilities, and accuracy on a geographical location, can enhance predictions. We believe that GPT 4 can integrate those information into a final response. One path is providing also INaturalist as possible functions to call, something we may test in the future.

We see a platform full of potential, especially, after all the advances in artificial intelligence in the last years, both in computer vision and conversational artificial intelligence. The best advances in AI was becoming public APIs, as so they became bricks for programmers, instead of final tools delivered already designed.

References

- Blank, S. and Dorf, B. (2012). *The Startup Owner's Manual: The Step-By-Step Guide for Building a Great Company*, K & S Ranch.
- Felipe Guimarães Dos Santos, Emanuel A. Machiaveli, J. M. V. M. (2021). Snake classifier: Aplicativo mobile para classificação de serpentes peçonhentas. https://www.academia.edu/95702218/Snake_Classifier_Aplicativo_mobile_para_classifica%C3%A7%C3%A3o_de_serpentes_pe%C3%A7onhentas.
- Ghosn, Y., Sardouk, O. E., Jabbour, Y., Jrad, M., Kamareddine, M. H., Abbas, N., Saade, C. and Ghanem, A. A. (2023). Chatgpt 4 versus chatgpt 3.5 on the final frcr part a sample questions. assessing performance and accuracy of explanations, *medRxiv*.
URL: <https://www.medrxiv.org/content/early/2023/09/08/2023.09.06.23295144>
- Jesse Palmer, Corinna Cohn, M. G. C. N. (2018). *Testing Angular Applications*, Manning Publications.
URL: <https://www.manning.com/books/testing-angular-applications>
- Khorikov, V. (2020). *Unit Testing Principles, Practices, and Patterns*, Manning Publications.
- Lubiana, T., Lopes, R., Medeiros, P., Silva, J. C., Gonçalves, A. N. A., Maracaja-Coutinho, V. and Nakaya, H. T. I. (2023). Ten quick tips for harnessing the power of chatgpt/gpt-4 in computational biology.
URL: <https://api.semanticscholar.org/CorpusID:257805086>

Pires, J. G. (2022). Innovating with biomathematics: the challenge of building user-friendly interfaces for computational biology, *Academia Letters* p. Article 5792.
URL: <https://doi.org/10.20935/AL5792>

Pires, J. G. (2023a). Data science using openai: testing their new capabilities focused on data science, *Qeios*.
URL: <https://www.qeios.com/read/76QMHB>

Pires, J. G. (2023b). Robodoc: a conversational-ai based app for medical conversations, *medRxiv* 2023: 2023.12–31.23300681.
URL: <https://www.medrxiv.org/content/10.1101/2023.12.31.23300681v1>

Pires, J. G. and Dias Braga, L. H. (2023). Snakeface: a transfer learning based app for snake classification, *Revista Brasileira de Computação Aplicada* 15(3): 80–95.
URL: <https://seer.upf.br/index.php/rbca/article/view/15028>

Rahaman, M. S., Ahsan, M. M. T., Anjum, N., Terano, H. J. R. and Rahman, M. M. (2023). From chatgpt-3 to gpt-4: A significant advancement in ai-driven nlp tools, *Journal of Engineering and Emerging Technologies*.
URL: <https://api.semanticscholar.org/CorpusID:258650483>

Rosoł, M., Gąsior, J. S., Łaba, J., Korzeniewski, K. and Młyńczak, M. (2023). Evaluation of the performance of gpt-3.5 and gpt-4 on the medical final examination, *medRxiv*.
URL: <https://api.semanticscholar.org/CorpusID:259073477>

West, C. G. (2023). Advances in apparent conceptual physics reasoning in gpt-4, *ArXiv abs/2303.17012*.
URL: <https://api.semanticscholar.org/CorpusID:257913307>