ORIGINAL PAPER

SnakeChat: a conversational-AI based app for snake classification

Jorge Guerra Pires [1]

[1]Founder at IdeaCoding Lab / JovemPesquisador.com, Brazil

*jorgeguerrapires@yahoo.com.br

Abstract

This document contains an extended method section and background for "SnakeChat: a conversational-AI based app for snake classification".

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

1 Background

1.1 Conversational Artificial Intelligence in biology

We have done a research using Semantic Scholar and Bing powered by chatGPT, and we were unable to find papers on chatbots in biology. Thus, to the best of our knowledge, we are the first researchers to apply openAI APIs to biology on the context of scientific research. However, we will still keep searching, and possibly publish a review paper if we find a considerable amount in time. The tendency is that they will be more and more common, as we are doing. Once the cost for implementing those CAI dropped, also, the entering barrier will also drop.

Building a CAI by itself would require a big team, but thanks to openAI, we can just use it, as we are doing herein; it is hard to imagine academic groups having access to such amounts of investments to build such a CAI. That could explain why we were unable to find papers on this topic: it would require a huge team to create the CAI, and then apply it to biological cases; the investment barrier was high. This is no longer the case. This is just one book on CAI before openAI APIs Freed (2021).

Which means, we are innovating. We also tried to find CAI in bionformatics, and the results are focused on medicine. We believe that we may spark researches on using those bots in biology, which could help on using those models in biology. Most models can be intimidating, and as we are going to see, we replace all the user interaction with SnakeFace by a chatbot, which we called SnakeChat.

Chatbots can be the UI/UX for the future of biology and medicine. We found on an informal survey that parametrization of models is a big barrier for medical doctors and biologists when using models Pires (2021).

1.2 A brief look at the literature on snake classification

Transfer learning applied to snake classifications is an important and promising technique in the field of machine learning Kalinathan et al. (2021); Mirunalini et al. (2022); Yang and Sinnott (2021); Progga et al. (2021); Lakshmi et al. (2021); Krishnan (2020). Several papers have explored this area, highlighting its potential for snake species identification and classification. SnakeFace Pires and Dias Braga (2023), a transfer learning based app, demonstrates the ease of building machine learning models without coding expertise and its adaptability to different biological image identification cases. Another paper proposes a deep learning model that uses transfer learning to build a snake species classifier, combining snake photographic images with geographic location information Desingu et al. (2021).

The problem of snake classification is the problem of assigning classes to snakes based on images. Those assignments can be such as their species. This is a problem of *computer vision*. This problem may have several applications, such as educational purpose, finding the proper venom serum. It is crucial to get the snake correctly, at least, with a certain level of accuracy.

Nonetheless, this problem may impose several challenges. We may have images that are not clear, or images that are not enough for forming a pattern on the training. In other cases, it possible that the necessary information is not on images. See Pires and Dias Braga (2023) for more discussions about SnakeFace, one of our base models.

One challenge we are going to face is that as the model becomes generic with more species on a single model, also, their precision on species starts to go down. However, as you have models with high accuracy, they become susceptible to bad usage (e.g., entering an image of a snake not present at the model). And this a trade-off hard to balance, as we are going to see,

Our model stands out on two main points: it was built in JavaScript and it was built to be used. The models we found on the literature are designed in Python Rajabizadeh and Rezghi (2021); Kalinathan et al. (2021); Yang and Sinnott (2021); Desingu et al. (2021); Durso AM and de Castañeda R (2021); Felipe Guimarães Dos Santos (2021), and they are not available for being easily used. This dominance of Python in machine learning is well-known, but challenged recently by JavaScript. There are several advantages of using JavaScript, the main one: web applications are built in JavaScript. Thus, one language for all the process: from machine learning to the app development. It may reduce for instance costs since you need just one language, and sometimes, even no server (e.g., Node.js or PHP).

If those models in Python were available as APIs, or any other easy form to be used (e.g., NPM packages), we would have been happy in comparing our findings with other models for snake classification. For so, we would need to implement one by one. It does not seem practical. Nonetheless, SnakeFace showed to be competitive with those models Pires and Dias Braga (2023). None of those models, to the best of our knowledge, explore chatbots. Chatbots can make it easier to use models, since all they require is a human-like command, and the artificial intelligence will handle the rest.

The reader can find a more detailed discussion about the literature on snake classification on the paper about SnakeFace Pires and Dias Braga (2023).

It is important to mention that INaturalist, from where we sampled most of our images of snakes, has their own open source app project, which is a general app. Their project is a community project. I have used their app for suggesting snakes: it tends to be good at associating similar snakes. Nonetheless, it is not good at actually make a final prediction. Initial prediction are later double-checked by humans, their community. The final prediction is a community consensus. Thus, it is an annotation process, more precisely, a crowd annotation. Their platform is perfect for us since the images are real, they are not professional images. The images is the type of images users may take when using our app, amateur images.

2 Methods Extended

We have used mainly two set of algorithms for creating SnakeChat: i) SnakeFace Pires and Dias Braga (2023); and ii) openAI APIs. SnakeFace is a transfer learning based set of models used to classify snakes focused on Brazil, it is a prototype, a concept testing. Whereas, openAI APIs is a set of APIs focused on large language models (LLMs), being chatGPT one of them.

On Section 2.1, it is explored the AI that has been used. We did our best effort to pass all the details necessary for reproducibility. In case of interest to reproduce, it would necessary learning more about the APIs before trying, we are assuming the reader knows the basic. A tutorial or just the official documentation must be used alongside.

On Section 2.2, parametrization of SnakeFace is discussed, and details considered important enough were reported .

On Section 2.3, it is presented further details, that could be essential to understand the paper.

Finally, on Section 2.4, links and external material are provided.

The reader is welcome to get in touch if any information is missing.

2.1 openAI APIs explored

2.1.1 *gpt-3.5-turbo-1106*

This is the latest version of the chatGPT API. It will be the default for chatGPT API in the future.

2.1.2 *gpt-4-1106-preview*

This is their most advanced model when we wrote this paper. It is their API for GPT 4, released after chatGPT; with vision capability and more complex reasoning. Several researchers are finding that this version is indeed superior to chatGPT, e.g. Ghosn et al. (2023).

2.1.3 gpt-4-vision-preview

This model is the vision behind GPT 4. When GPT 4 was released, their vision capability was largely commented upon. It can interpreter complex images and get meaning from them. As one example, we have also tested in collaboration with another author this algorithm applied to medical images.

> "The image you've provided is a dental X-ray, also known as a radiograph. This type of image is used by dentists to evaluate the health of teeth and the surrounding structures that are not visible during a regular dental examination. In this X-ray, we can see several teeth with their roots. The image shows the internal structure of the teeth including the pulp chambers and canals. There's also visibility of the surrounding bone. Dentists examine these images for various purposes, such as checking for cavities between teeth, looking at tooth roots, assessing the health of the bony area around the tooth, and checking the status of developing teeth. However, without being a trained dental professional and without more context, it's not appropriate to comment on the specifics regarding the health of these teeth or diagnose any conditions." *gpt-4-vision-preview* reading a medical X-ray

Sadly, as we have discussed with a medical doctor also doing research on those tools, it cannot make a precise diagnosis. It is similar to snakes: it cannot pinpoint precisely the snake's species. What we have learnt is that coupling those models with a more precise model can create something quite advanced. They have released what is called GPTs. It can be trained to better read images; the authors discussed with another author training those GPTs, and the results are promising. Nonetheless, we are going to follow a different step: we are going to use an external library, namely, SnakeFace, which is based in TensorFlow.js.

Table I: Different settings for SnakeChat using openAI APIs for Table II.

Section	vision	model selector	final response
section 1	*gpt-4-vision-preview*	*gpt-3.5-turbo-1106*	*gpt-3.5-turbo-1106*
section 2	*gpt-4-vision-preview*	*gpt-4-1106-preview*	*gpt-3.5-turbo-1106*
section 3	*gpt-4-vision-preview*	*gpt-4-1106-preview*	*gpt-4-1106-preview*

2.1.4 Table of results

We have tested combinations of the algorithms we just mentioned Table II. The goal is presenting how the algorithm will possibly behavior with different configurations. This can be easily added as a configuration option from the SnakeChat, with their respective costs.

See Supplementary Material for more examples, using more configurations.

2.2 Parametrization of the algorithms

2.2.1 Parametrization of SnakeFace

We have used SnakeFace basically as reported on Pires and Dias Braga (2023); differences in behaviors may exist since we had to retrain the models when writing the paper about SnakeFace, but we have not retrained the models now, when developing SnakeChat; we have just uploaded locally the pretrained models. A pretrained model is a neural network model used after training; it is a model with knowledge. ChatGPT is a pretrained model, as it is MobileNet largely used for image identification.

The models used here are below. Those links are used to upload the pretrained models locally, this is SnakeFace when we mention it: a set of pretrained models trained to identify snakes. See Pires and Dias Braga (2023) for more details. We should stress that those models are prototypes. A production model should have much more snakes to be interesting enough.

i. Fake coral snakes (model 1): `https://teachablemachine.withgoogle.com/models/W9_qIu14Y/`
ii. Fake vs. true coral snakes model (model 2): `https://teachablemachine.withgoogle.com/models/9vw2M7LJw/`
iii. Model with variety (model 3): `https://teachablemachine.withgoogle.com/models/Sc8mKQsS0/`

Tip. if you click on the link, the model will open on the browser and you can input images.

We are going to use *gpt-3.5-turbo-1106* to automatically decide which models to call; later, we shall also try out *gpt-4-1106-preview*, the GPT 4 API. This is called function calling by the openAI team. This function is interesting since you give functions for the chat model to decide whether it should look for external information. For example, if you ask the temperature in a city, and the model has access to an API about temperature, it can call the API, get the temperature, and give back an educated response; instead of making it up as pure chatGPT famously do.

This approach of connecting chatGPT with powerful external tools has been largely explored. For example, the group behind Matematica, a famous numerical software, used this approach as so chatGPT wil give educated responses.

2.2.2 Parametrization of openAI APIs

1st step: getting a description from the image

Bellow is the section of the code that calls openAI for getting an image description. It is coded in Typescript and used in Angular. One point: the user can provide a human description as alternative. The AI generated description is triggered just when no description is provided by the user. It was also tested using MobileNet, which is a general-purpose computer vision model, for getting a general view of the image (see SM).

```
//It will return a promise, with a call to openAI
return await this.openai.chat.completions.create({
  max_tokens:300,//size of reponse, in tokens
  model: "gpt-4-vision-preview",//model used on the call
  messages: [//Message sent
    {
      role: "user",
      content: [
        { type: "text",
        text:
        'There is a snake on this image.
        Describe it.
        Focus on the colors and patterns.
        Ex. the snake is black, red, and white.
        The colors are in strip patterns.
        The black strips appear more easily than
        the red ones.' },
        {
          type: "image_url",
          image_url: {
            "url": this.selectedImage,
          },
        },
      ],
    },
  ],
}).then((response)=>{
   //once the promised is finished
  //getting just the response
  return response.choices[0].message.content;
})
```

2nd step: getting which models from SnakeFace to call

The process of calling openAI API is the same as previously. The only change is the parametrization.

```
  const options:any = {
  model: "gpt-3.5-turbo-1106",
  messages: messages,
  tools: tools,
  tool_choice: "auto",
};
```

gpt-3.5-turbo-1106 is their basic model, it has all the basic chatGPT capability, and with a large context (the size of possible inputs to the model, basic chatGPT is about 2 pages).

tools are the SnakeFace functions explained as so the model knows what each model does. Explanation given:

i. Fake coral snakes (model 1): Should be used only on snakes that look like coral snakes. This model was trained to identify fake coral snakes: snakes that look like coral snakes. The response is the snake with highest probability: name and the probability;
ii. Fake vs. true coral snakes model (model 2): Should be used only on snakes that looks like coral snakes. This

model was trained to identify both fake and true coral snakes: snakes that look like coral snakes. The response is the snake with highest probability: name and the probability;

iii. Model with variety (model 3): This model was trained to identify several snakes. It is a generic model for classifying snakes from different species. The response is the snake with highest probability: name and the probability.

It is important to stress that those texts are called *prompt engineering*. Most of the success of using the openAI API is testing those texts. Changes may happen. Generally, we try commands, then we change looking for desired behaviors. We are constantly changing those texts, looking for better behaviors from the API, it may change in the future.

The links to the models were given previously, and used to call them.

3rd step: gathering all the response, and getting a conclusion

This is the most tricky part. Several scenarios may happen. For instance, the SnakeFace prediction may not be in accordance with the openAI API suggestion. This is where those possible conflicting information may happen. We are constantly testing ways to handle those possible undesirable scenarions.

On this last part, we use again *gpt-3.5-turbo-1106* for creating a final response. On the final information sent to *gpt-3.5-turbo-1106*, we have a JSON response from SnakeFace.

```
this.openai.chat.completions.create({
          model: "gpt—3.5—turbo—1106",
          messages: messages,
        }).then((secondResponse)=>{

          this.message_chat.push(
          ‘Em conclus o:
          \n ${secondResponse.choices[0].message.content} ‘);
        }); e
```

Please, find it here the complete method as Gist (a GitHub file of code).

2.3 Further details

2.3.1 *Using Teachable Machine*

SnakeFace was trained using Teachable Machine Pires and Dias Braga (2023).

We wanted to also experiment with models that have no overlapping: no overlapping between the species on the models. For the models originally published by Pires and Dias Braga (2023), there are overlapping between the models: model 2 contains model 1, and model 3 contains partially model 1 and model 2. On their scenario of research, it made sense such superposition.

For the case discussed herein, it becomes an issue. The same apporach from Pires and Dias Braga (2023) was used for our models. Model 1 is the same, except we have retrained the model. We have discarded models 2 and 3 for having overlapping; but their results are still on the paper (Table II). They were replaced with a model just for true coral snakes, and one model just for *bothrops* genus. The species are the same, we have just reorganized them into separate sub-models, and those sub-models have no species in common. We wanted to investigate the trade-off between large models and specialized models on our chatbot.

The training graphs were added as Supplementary Material.

2.3.2 *Model validation*

The validation of this system is tricky. Validation is the process by which we make sure our system can achieve its goal, on a statistical meaningful level. One possible validation for our case is seeing how well it classifies snakes. The classification is done in three steps: i) get an image textual description (which is provided automatically by openAI API in case it is not provided by the user); ii) classify by image classification (SnakeFace); iii) use the information so far for making a final guess (openAI API).

The system is composed of two parts: SnakeFace and openAI API. Therefore, the first step is validating them. SnakeFace was validated on our other paper Pires and Dias Braga (2023) and also on the SM, whereas the openAI API is validated on their respective official documentation, we provide links to the documentation when we use the models. Therefore, the validation of the parts are not our concern, it is out of scope. We assume they were properly done.

The second part that could be validation is how the system works as a whole, the sum of the parts: how the image classification provided by SnakeFace is supported by the openAI API, which is our goal. We believe it is meaningless this validation, we shall not do it. The reason is: a single change on either of the APIs we are using will make changes on the SnakeChat behavior: making the validation outdated. OpenAI APIs are constantly updated, and SnakeFace is being developed. Thus, the best we can do is trusting their respective validations.

Our validation is seeing how the system behavior, which we did at the discussion section. We have also provided a SM material for several real conversations with comments and comparing with other alternatives. That is the best validation we believe we can do, that makes sense for SnakeChat.

2.3.3 *Inputting images*

The current model is built to be used, by a non-expert user. The model is built on top of other models/APIs. Any image that could be uploaded to a browser will be just fine for classification; you can try to upload the image on our prototype, if the upload appears to you on the browser, it is fine. All the necessary image preprocessing is either done by us or by third-party APIs.

The ideal image is a clear image. If you add a non-clear image (e.g., Fig. 1), the model may get confused, and misclassify; in some cases, the misclassify can be bad, like classifying *Bothrops alternatus* as *micrurus corallinus*. See SM for another example of image that is hard to classify. Our hope is that GPT 4 with their description capability will support on this kind of mistakes. GPT 4 Vision can see and describe Fig. 1 properly. Interesting enough, SnakeFace, which uses transfer learning, cannot see very well colors. See SnakeFace for more details Pires

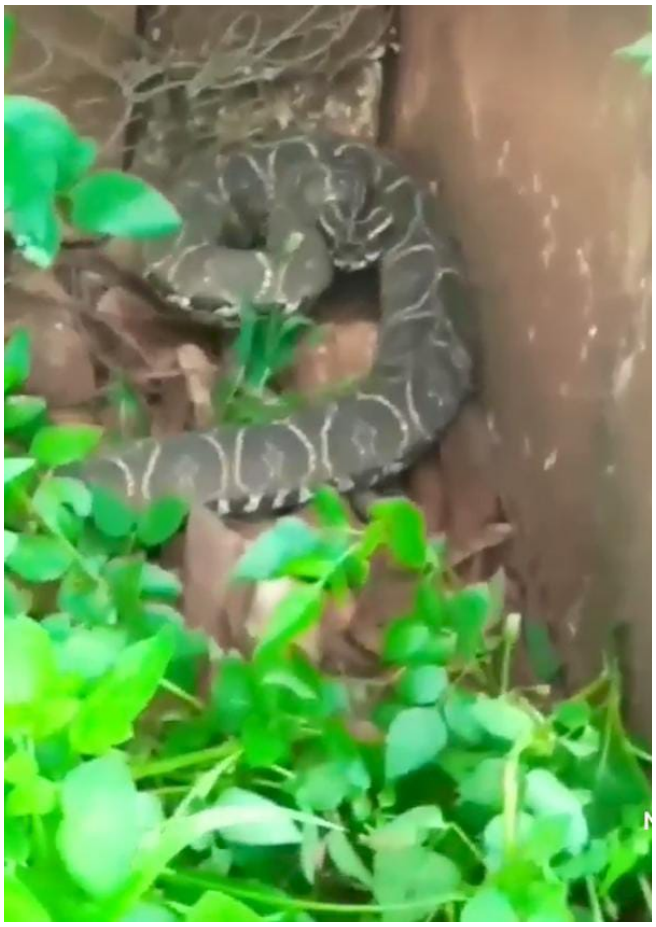

Figure 1: A *bothrops alternatus*, hard to classify. See that during the training, it was possible to classify this image, but as we added new images/species, we lost this good result. The goodness of the model is measured globally, not for a specific image or species. See Pires and Dias Braga (2023) for more details.
. Source: Pires and Dias Braga (2023).

and Dias Braga (2023), once this is the model that makes the image classification. Regarding the image description by GPT 4, see their documentation for more details.

2.3.4 *Transfer learning*

Our model is using models that are based on *transfer learning*. How those models are built is discussed on the Method section of Pires and Dias Braga (2023). On this aforementioned paper, it is possible to see the performance metrics for the models. The models that have been used herein were retrained, and most likely it will be retrained as we add new species, and run new experiments. Therefore, it is essential to focus on the qualitative behavior (e.g., whether the accuracy is higher enough and the loss function is low alongside the testing curves), not the quantitative (e.g., whether we get a predefined number on accuracy). Once the models are retrained, it is focused on the overall behavior. See Pires and Dias Braga (2023) for getting familiar with how those models are trained, and what to expect from those training procedures. On the case presented, it was used models already trained.

Figure 2: SnakeFace logo.

The chatbot that is discussed herein does not depend on the model used under the hood, neither it depends whether a specific technique is used (e.g., transfer learning). Therefore, it was made the decision not to focus on how the models under the hood were trained, it is discussed on Pires and Dias Braga (2023) and the SM. The reader is gently refereed to this paper in case of interest on the models under the hood, which is out of focus of the current paper. Any model can be added under the hood, instead of the ones that were used. Any metrics presented herein would be most likely outdated soon since it is planned to add new species and test new approaches, which shall be discussed on future publications. Any new results will not affect the current paper, since the chatbot uses the models, but it does depend on those models for working.

One interest fact about our approach: "the brain" (i.e., the models) is separated from the interface (i.e., the chatbot).

2.4 External resources: getting to know our prototype

In addition to the links already provided, you can also access a version of the model deployed here. You can upload an image and test it yourself.

Currently, we can classify these species, see SnakeFace for more details Pires and Dias Braga (2023). When we use the openAI API, we are shifting between those models. The task of the openAI API is creating a conversation, and call the proper model for answering the question; currently, you just upload the image, and the model will decide which model(s) to call, and add some possible context. If you upload an image outside of the list, the model will try to fit on this list: it will take the highest one.

Please, for more details, see Pires and Dias Braga (2023). Also, if you upload a no-snake image, it will still classify. We have tested on SM using MobileNet for avoiding these undesirable behavior. It can be solved adding a filter on the image workflow, which will avoid no-snake images entering the classification workflow; MobileNet can do it without any need of training. That something we intend to work on in the future, it is straightforward to implement (see SM for some initial testing).

2.4.1 model 1: just fake coral snakes

Namely: 1) *Apostolepis assimilis* ; 2) *Erythrolamprus aesculapii* ; 3) *Oxyrhopus rhombifer* ; 4) *Lampropeltis triangulum triangulum*.

See SM, this was modified on the current format. *Lampropeltis triangulum triangulum* was eliminated for not being concentrated on our geographical region of interest.

2.4.2 model 2: false coral snakes vs. true ones

On this model, we are using the following true coral snakes: 1) *Micrurus corallinus* ; 2) *Micrurus frontalis* ; 3) *Micrurus alleni*; 4) *Micrurus albicinctus* . For the false ones: 1) *Apostolepis assimilis*; 2) *Erythrolamprus aesculapii* ; 3) *Oxyrhopus rhombifer* .

2.4.3 model 3: experimenting with diversity

For the fake coral snakes: 1) *Apostolepis assimilis*; 2) *Erythrolamprus aesculapii*; 3) *Oxyrhopus rhombifer*.

And for the diversity, we have used: 1) *Crotalus durissus* ; 2) *Bothrops alternatus* ; 3) *Bothrops neuwiedi* ; 4) *Bothrops jararaca*. Since we already know the difficulties between false and true coral snakes, we left the true coral snakes out.

On the latest model, see SM, we have added also the true coral snake on this model.

References

Desingu, K., Mirunalini, P. and Kumar, J. (2021). Snake species classification using transfer learning technique, *Conference and Labs of the Evaluation Forum*.
URL: *https://api.semanticscholar.org/CorpusID:237298537*

Durso AM, Moorthy GK, M. S. B. I. S. M. and de Castañeda R, R. (2021). Supervised learning computer vision benchmark for snake species identification from photographs: Implications for herpetology and global health, *Frontiers in Artificial Intelligence* **4**. https://doi.org/10.3389/frai.2021.582110.

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.

Freed, A. (2021). *Conversational AI: Chatbots that work*, Manning.
URL: *https://www.amazon.com/Conversational-AI-Chatbots-that-work/dp/1617298832*

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*

Kalinathan, L., Balasundaram1, P., Ganesh, P., Bathala, S. S. and Mukesh, R. K. (2021). Automatic snake classification using deep learning algorithm, *Conference and Labs of the Evaluation Forum*, . https://ceur-ws.org/Vol-2936/paper-135.pdf.

Krishnan, M. G. (2020). Impact of pretrained networks for snake species classification, *Conference and Labs of the Evaluation Forum*.
URL: *https://api.semanticscholar.org/CorpusID:225073809*

Lakshmi, D., chandra Panda, R., Amrita and Prakash, A. (2021). Life-saving app: Snake classification 'venomous and non-venomous' using fast.ai based on indian species, *Artificial Intelligence Systems and the Internet of Things in the Digital Era* .
URL: *https://api.semanticscholar.org/CorpusID:236671950*

Mirunalini, P., Desingu, K., Bharathi, H., Chodisetty, E. A. and Bhaskar, A. (2022). Deep learning and gradient boosting ensembles for classification of snake species, *Conference and Labs of the Evaluation Forum*.
URL: *https://api.semanticscholar.org/CorpusID:251470924*

Pires, J. G. (2021). An informal survey presents the gap between computer and medical doctors and biologists, Medium. Accessed on 6 December 2023.
URL: *https://medium.com/theoretical-and-mathematical-biology/an-informal-survey-presents-the-gap-between-computer-and-medical-doctors-and-biologists-ca8816051739*

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*

Progga, N. I., Rezoana, N., Hossain, M. S., ul Islam, R. and Andersson, K. (2021). A cnn based model for venomous and non-venomous snake classification, *Analogical and Inductive Inference*.
URL: *https://api.semanticscholar.org/CorpusID:236963195*

Rajabizadeh, M. and Rezghi, M. (2021). A comparative study on image-based snake identification using machine learning, *Scientific Reports* **11**(1). https://doi.org/10.1038/s41598-021-96031-1.

Yang, Z. and Sinnott, R. O. (2021). Snake detection and classification using deep learning, *Proceedings of the 54th Hawaii International Conference on System Sciences*, Proceedings of the 54th Hawaii International Conference on System Sciences, pp. 1212–1222. https://scholarspace.manoa.hawaii.edu/server/api/core/

bitstreams/f9f2de4d-ddaa-4984-9fc3-9bbca4e27247/
content.