TransferLab

Testing with Concept Activation Vectors (TCAV)

Ivan Rodriguez

October 26, 2023

Kim et. al., “Interpretability beyond feature attribution: Quantitative Testing with Concept Activation Vectors (TCAV)”. In ICML - PMLR (2018). github

What is TCAV ?

- **TCAV is an XAI method with the goal of making the outcomes of your model reliable and trustworthy for non-technical end users, such as doctors.**

What is TCAV ?

- **TCAV is an XAI method with the goal of making the outcomes of your model reliable and trustworthy for non-technical end users, such as doctors.**

- **TCAV introduces the notion of “concepts” in a model's prediction. For example, it can explain how the concept of “striped” influences a model's classification of an image as a “zebra.**

What is TCAV ?

- TCAV is an XAI method with the goal of making the outcomes of your model reliable and trustworthy for non-technical end users, such as doctors.
- TCAV introduces the notion of “concepts” in a model's prediction. For example, it can explain how the concept of “striped” influences a model's classification of an image as a “zebra.
- TCAV describes the relationship between a concept and a class rather than explaining a single prediction, providing valuable global interpretation of a model’s behavior.

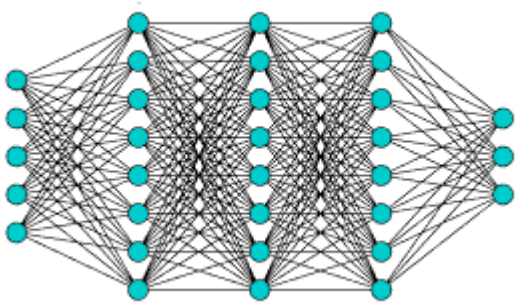
What is TCAV ?

- TCAV is an XAI method with the goal of making the outcomes of your model reliable and trustworthy for non-technical end users, such as doctors.
- TCAV introduces the notion of “concepts” in a model's prediction. For example, it can explain how the concept of “striped” influences a model's classification of an image as a “zebra.
- TCAV describes the relationship between a concept and a class rather than explaining a single prediction, providing valuable global interpretation of a model’s behavior.
- TCAV is primarily applied to DNNs, but it has the potential to work with any model where calculating directional derivatives is feasible (more clarity on this later).

TCAV idea

Given a trained DNN (regressor or classifier):

l

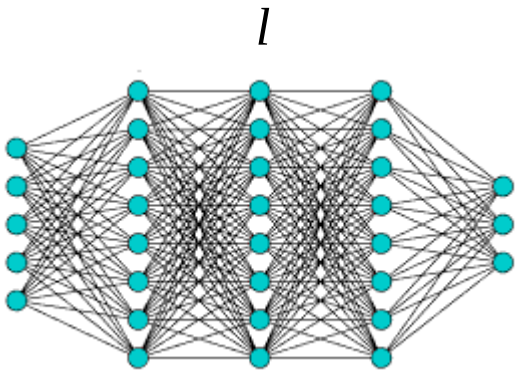


Devise a method to measure how sensitive is a given layer, l, to a certain concept that could describe our data.

For example the concepts “stripped”, “zigzaged” or “dot” for a dataset containing images of zebras.

TCAV idea

Given a trained DNN (regressor or classifier):



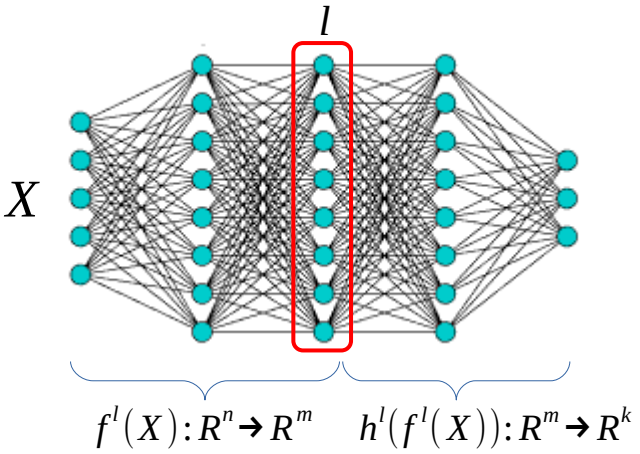
Devise a method to measure how sensitive is a given layer, l, to a certain concept that could describe our data.

For example the concepts “stripped”, “zigzaged” or “dot” for a dataset containing images of zebras.

The paper proposes a simple and interesting approach through what they call:

Concept Activation Vector (CAV)

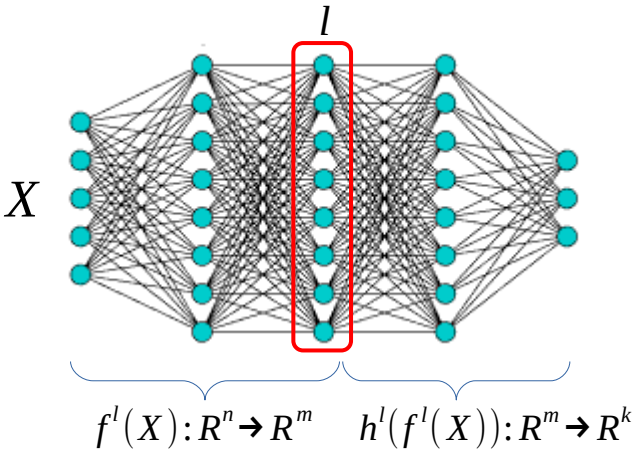
Concept Activation Vector (CAV)



Recipe to create a CAV

1 – select layer l with output $f^l(X) \in R^m$

Concept Activation Vector (CAV)



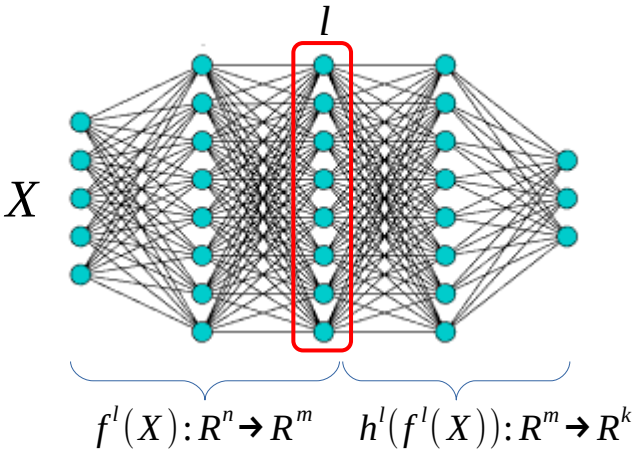
Recipe to create a CAV

1 – select layer l with output $f^l(X) \in R^m$

2 – Create a concept dataset, e.g concept ‘striped’ ➡ Set-A = {  }

(don’t need to belong to your data)

Concept Activation Vector (CAV)



Recipe to create a CAV

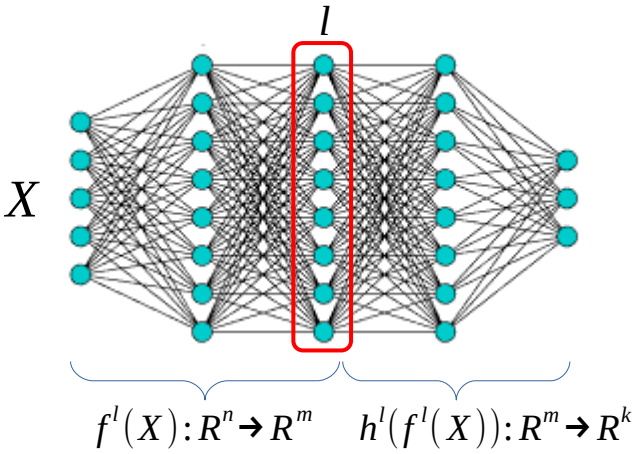
1 – select layer l with output $f^l(X) \in R^m$

2 – Create a concept dataset, e.g concept ‘striped’ ➡ Set-A = 

(don’t need to belong to your data)

3 – Generate a random dataset from your data ➡ Set-B = 

Concept Activation Vector (CAV)



Recipe to create a CAV

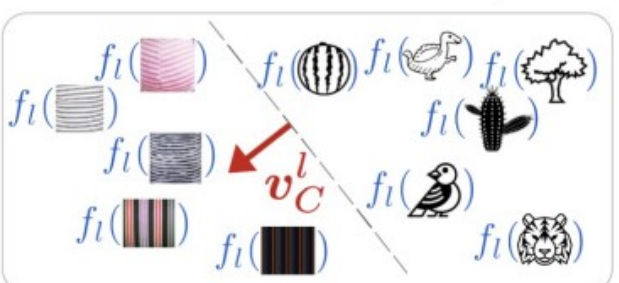
1 – select layer l with output $f^l(X) \in R^m$

2 – Create a concept dataset, e.g concept ‘striped’ ➡ Set-A = { }

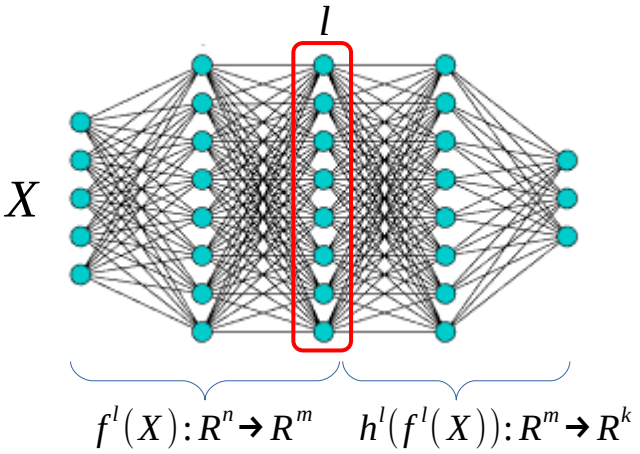
(don’t need to belong to your data)

3 – Generate a random dataset from your data ➡ Set-B = { }

4 – Train a linear binary classifier:
Set-A: class-0 and Set-B: class-1 .



Concept Activation Vector (CAV)



Recipe to create a CAV

1 – select layer l with output $f^l(X) \in R^m$

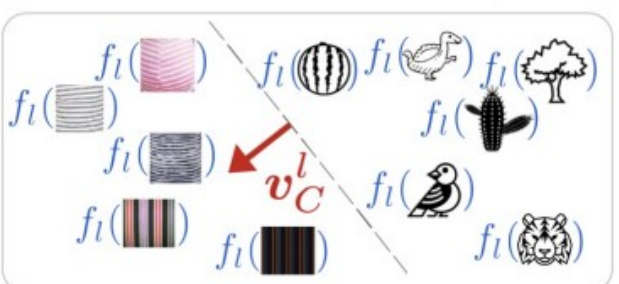
2 – Create a concept dataset, e.g concept ‘striped’ ➡ Set-A = { }

(don’t need to belong to your data)

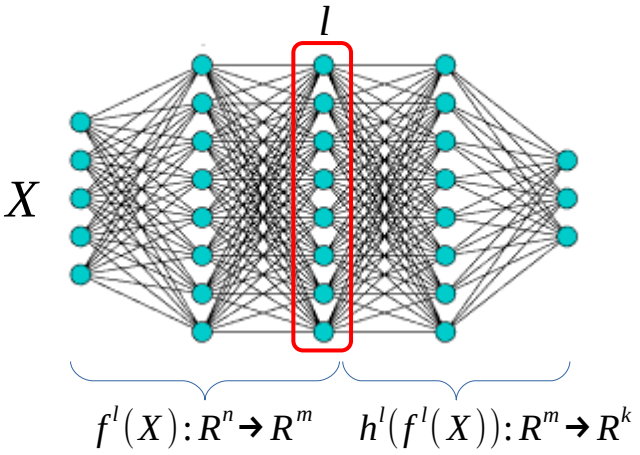
3 – Generate a random dataset from your data ➡ Set-B = { }

4 – Train a linear binary classifier:
Set-A: class-0 and Set-B: class-1 .

5 – $\boldsymbol{v}_C^l$ **is the CAV**



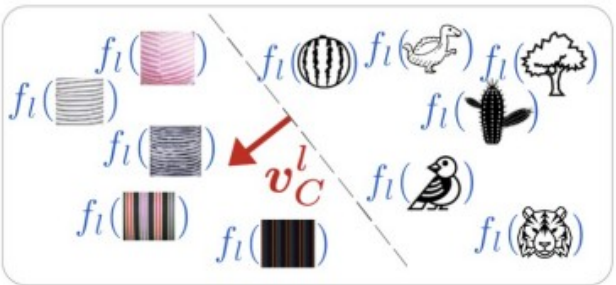
Concept Activation Vector (CAV)



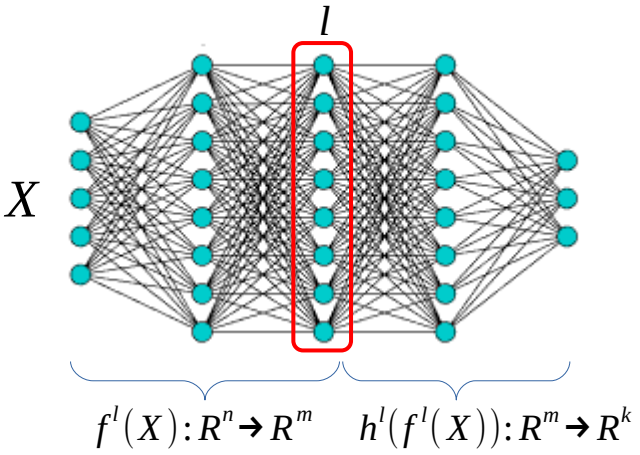
Recipe to create a CAV

5 – v_C^l is the CAV

6 – Conceptual sensitivity $S_{C,k,l} = \nabla h_k^l(f^l(X)) \cdot v_C^l$



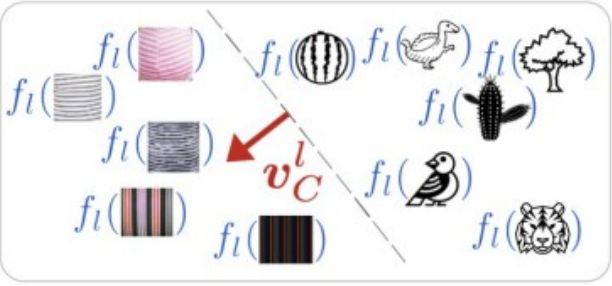
Concept Activation Vector (CAV)



Recipe to create a CAV

5 – v_C^l is the CAV

6 – Conceptual sensitivity $S_{C,k,l} = \nabla h_k^l(f^l(X)) \cdot v_C^l$



$S_{C,k,l}$: Sensitivity of class k to the concept C learned in the layer l .

In other words how important is the concept C in layer l for predicting the class k

TCAV Score

Let k be a class label for a given supervised learning task and let X_k denote all inputs with that given label. We define the TCAV score to be:

$$TCAV_{Q_{C,k,l}} = \frac{\left|\{x \in X_k : S_{C,k,l}(x) > 0\}\right|}{|X_k|}$$

with $$S_{C,k,l} = \nabla h_k^l(f^l(X)) \cdot v_C^l$$

TCAV Score

Let k be a class label for a given supervised learning task and let X_k denote all inputs with that given label. We define the TCAV score to be:

$$TCAV_{Q_{C,k,l}} = \frac{\left|\{x \in X_k : S_{C,k,l}(x) > 0\}\right|}{|X_k|}$$

with $$S_{C,k,l} = \nabla h_k^l(f^l(X)) \cdot v_C^l$$

Note that $TCAV_{Q_{C,k,l}}$ considers only the sign of $S_{C,k,l}$

TCAV Score

Let k be a class label for a given supervised learning task and let X_k denote all inputs with that given label. We define the TCAV score to be:

$$TCAV_{Q_{C,k,l}} = \frac{\left|\{x \in X_k : S_{C,k,l}(x) > 0\}\right|}{|X_k|}$$

with $$S_{C,k,l} = \nabla h_k^l(f^l(X)) \cdot v_C^l$$

Note that $TCAV_{Q_{C,k,l}}$ considers only the sign of $S_{C,k,l}$

For instance if the class k =“zebra” and C=”striped”: TCAV will measure how important is for layer l the concept striped for the whole class.

TCAV Score

Let k be a class label for a given supervised learning task and let X_k denote all inputs with that given label. We define the TCAV score to be:

$$TCAV_{Q_{C,k,l}} = \frac{\left|\{x \in X_k : S_{C,k,l}(x) > 0\}\right|}{|X_k|}$$

with $$S_{C,k,l} = \nabla h_k^l(f^l(X)) \cdot v_C^l$$

Note that $TCAV_{Q_{C,k,l}}$ considers only the sign of $S_{C,k,l}$

For instance if the class k =“zebra” and C=”striped”: TCAV will measure how important is for layer l the concept striped for the whole class.

You can use $TCAV_Q$ to detect biases in datasets. For example, if you're classifying sports, you could create concepts like "man" and "woman" and calculate $TCAV_Q$ for the entire dataset. The resulting score will help you see if there is a bias in the data.

Relative TCAV

What if there are correlation between the concepts, e.g. brown hair vs. black hair, in that case $v^l_{C_1}, v^l_{C_2}$ are far from orthogonal.

Relative TCAV

What if there are correlation between the concepts, e.g. brown hair vs. black hair, in that case $v^l_{C_1}, v^l_{C_2}$ are far from orthogonal.

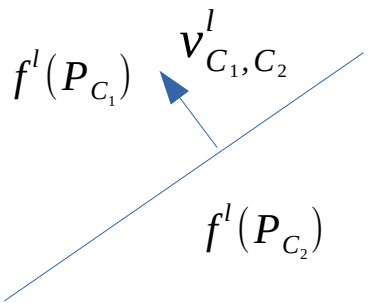
Given a class with high $TCAV_Q$ on both concepts (black and brown hair) it is possible to decide if one of them is more important than the other, for the given class? .

Relative TCAV

What if there are correlation between the concepts, e.g. brown hair vs. black hair, in that case $v^l_{C_1}, v^l_{C_2}$ are far from orthogonal.

Given a class with high $TCAV_Q$ on both concepts (black and brown hair) it is possible to decide if one of them is more important than the other, for the given class? .

Given the two concept sets P_{C_1}, P_{C_2}

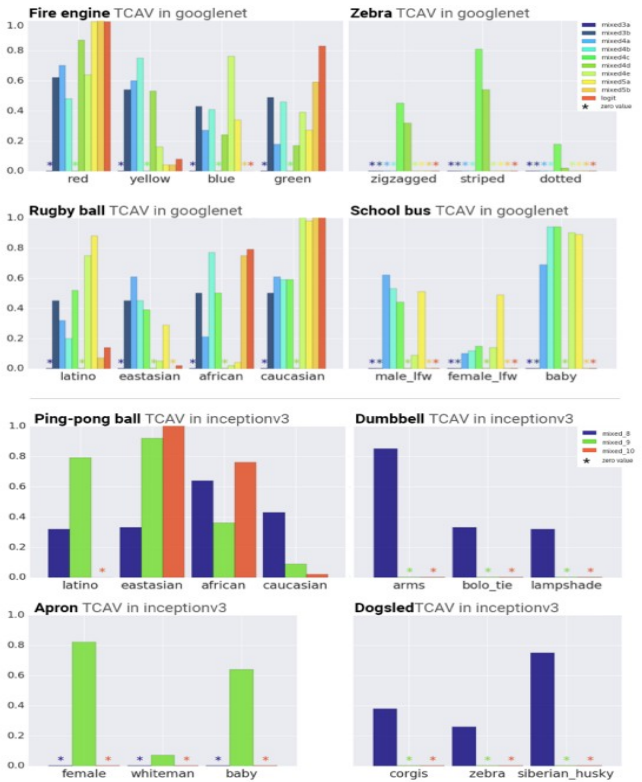


Use previous projection but with $v^l_{C_1,C_2}$

Results

1 - Various types of concepts, including color, texture, objects, gender and race.

2 - TCAV results with CAVs learned from all (for GoogleNet) or a subset (for Inception V3) of layers.



The results make sense:

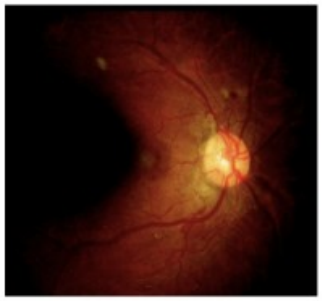
- Ping-pong balls highly correlated with particular race concept .

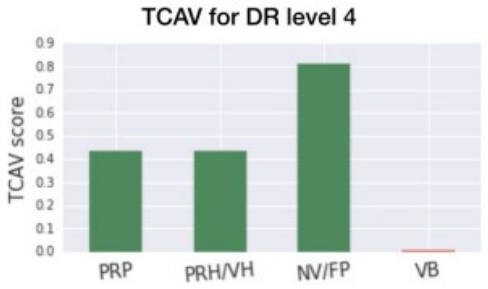
- Female concept highly relevant to the ‘apron’ class.

- Texture concepts influences $TCAV_Q$ in early layers (as expected in CNN). The opposite is observed for more complex concepts .

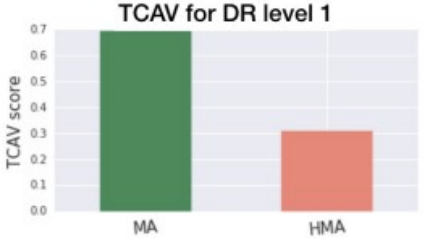
Results

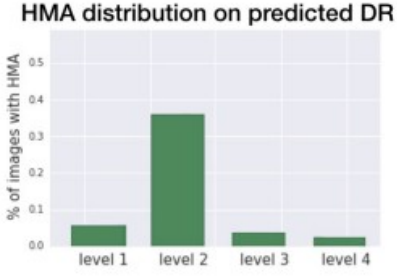
DR level 4 Retina





DR level 1 Retina



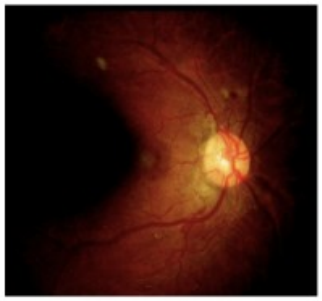


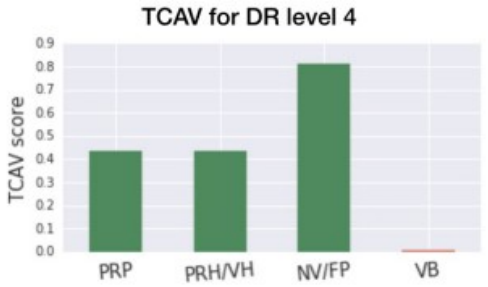
Diabetic Retinopathy (DR) data.

The model of interest predicts DR level using a 5-point grading scale based on complex criteria, from level 0 (no DR) to 4 (proliferative) .

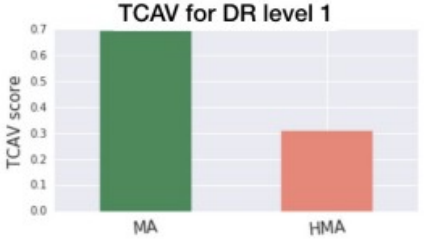
Results

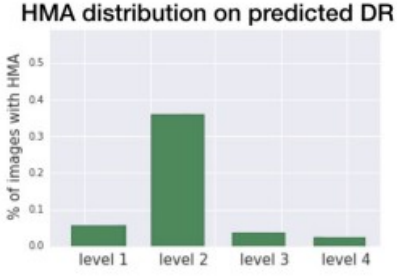
DR level 4 Retina





DR level 1 Retina





Diabetic Retinopathy (DR) data.

The model of interest predicts DR level using a 5-point grading scale based on complex criteria, from level 0 (no DR) to 4 (proliferative) .

For level 4 the model performance is good and the concepts (histogram) are in agreement with the expert expectations.

For level 1 the model performance is not so good and there is a missclassification of level 1 by level 2. The concept MA is also more relevant to level 2 so the concept is also wrong as expected,

Pros:

1 – Machine learning expertise not needed to employ TCAV

2 – Allow users to investigate any concept defined by a concept dataset.

3 – Provide global explanations, linking concepts to any class.

4 – Could help to identify potential issues or biases in model training: e.g. if a classifier is more sensitive to the concept “man” rather than “woman” it could suggest an imbalanced dataset.

Cons:

1 – Requires additional data for concept datasets, so it can be quite expensive.

2 – TCAV less applicable in shallow networks: deeper layers tend to offer better separability of concepts.

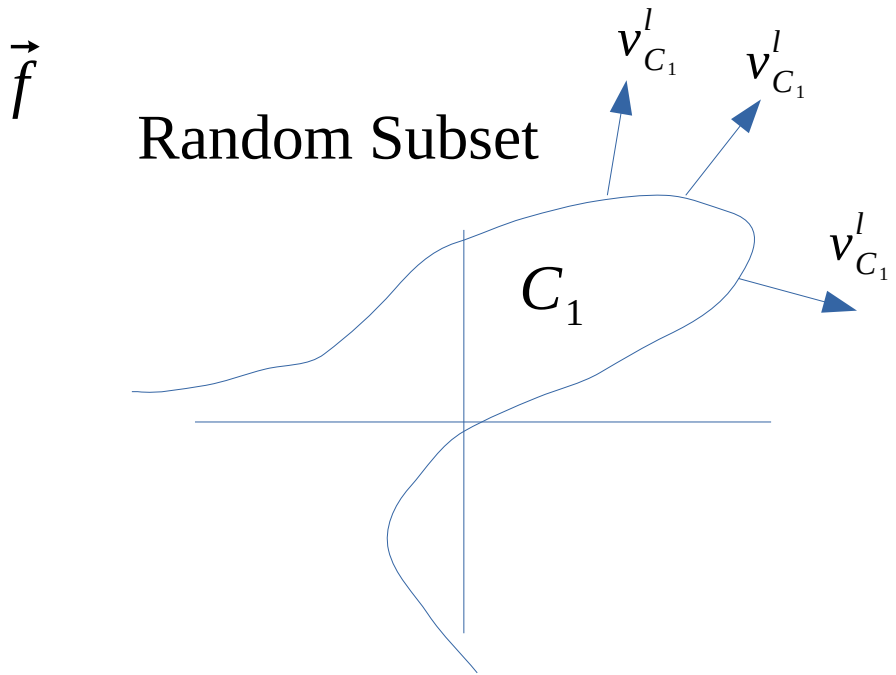
3 - Challenge with abstract or general concepts, e.g. "happiness". On those cases a fair amount of data is needed to train a Concept Activation Vector (CAV).

4 – Too fragile to adversarial examples.

Some thoughts:

1 – TCAV only uses the sign of the sensitivity: why not the magnitude of the projection too?

2 – Why a linear classifier? Can we tackle the problem with a non-linear one?
In this case you could take into account the correlations between different concepts.

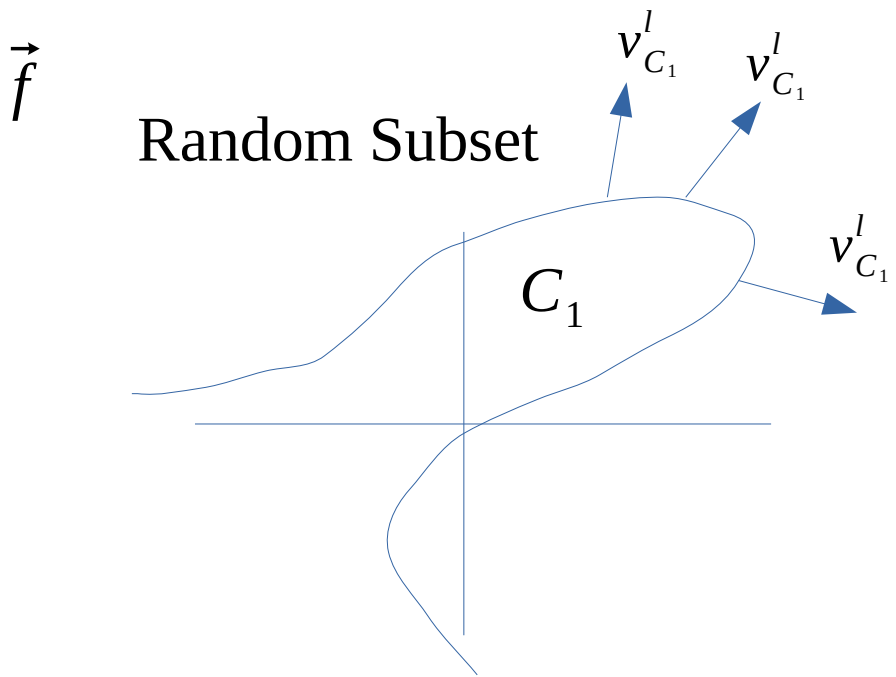


Non-linear classifier

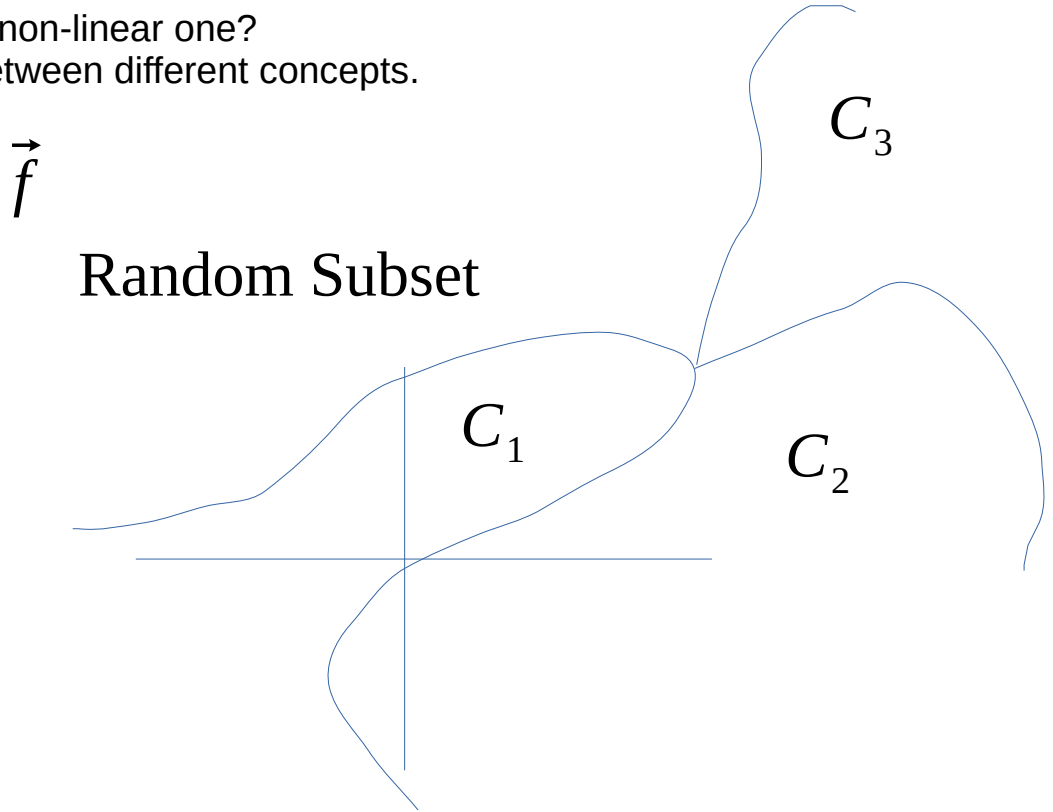
Some thoughts:

1 – TCAV only uses the sign of the sensitivity: why not the magnitude of the projection too?

2 – Why a linear classifier? Can we tackle the problem with a non-linear one?
In this case you could take into account the correlations between different concepts.



Non-linear classifier



Non-linear classifier with overlapping concepts

Concept Bottleneck Models (CBM)

This is another approach to introduce high-level concepts into the model.

Koh. et. al. - “Concept Bottleneck Models”, ICML (2020) . Github

Concept Bottleneck Models (CBM)

This is another approach to introduce high-level concepts into the model.

However, in contrast to TCAV, the idea is that we can intervene on these concepts. This means that:

a – By editing their predicted values and propagating these changes to the final prediction, we can often improve the model's accuracy during inference!

Koh. et. al. - "Concept Bottleneck Models", ICML (2020) . Github

Concept Bottleneck Models (CBM)

This is another approach to introduce high-level concepts into the model.

However, in contrast to TCAV, the idea is that we can intervene on these concepts. This means that:

a – By editing their predicted values and propagating these changes to the final prediction, we can often improve the model's accuracy during inference!

b – We can create counterfactual explanations, such as: if the model did not detect a bone spur (with 'bone spur' as the concept) in the x-ray, would it still predict severe arthritis?

Koh. et. al. - "Concept Bottleneck Models", ICML (2020) . Github

Concept Bottleneck Models (CBM)

This is another approach to introduce high-level concepts into the model.

However, in contrast to TCAV, the idea is that we can intervene on these concepts. This means that:

a – By editing their predicted values and propagating these changes to the final prediction, we can often improve the model's accuracy during inference!

b – We can create counterfactual explanations, such as: if the model did not detect a bone spur (with 'bone spur' as the concept) in the x-ray, would it still predict severe arthritis?

One of the main differences (and **disadvantages**) with TCAV is is the **requirement for concept annotation**.

Koh. et. al. - "Concept Bottleneck Models", ICML (2020) . Github

Let’s give an overview to the main idea

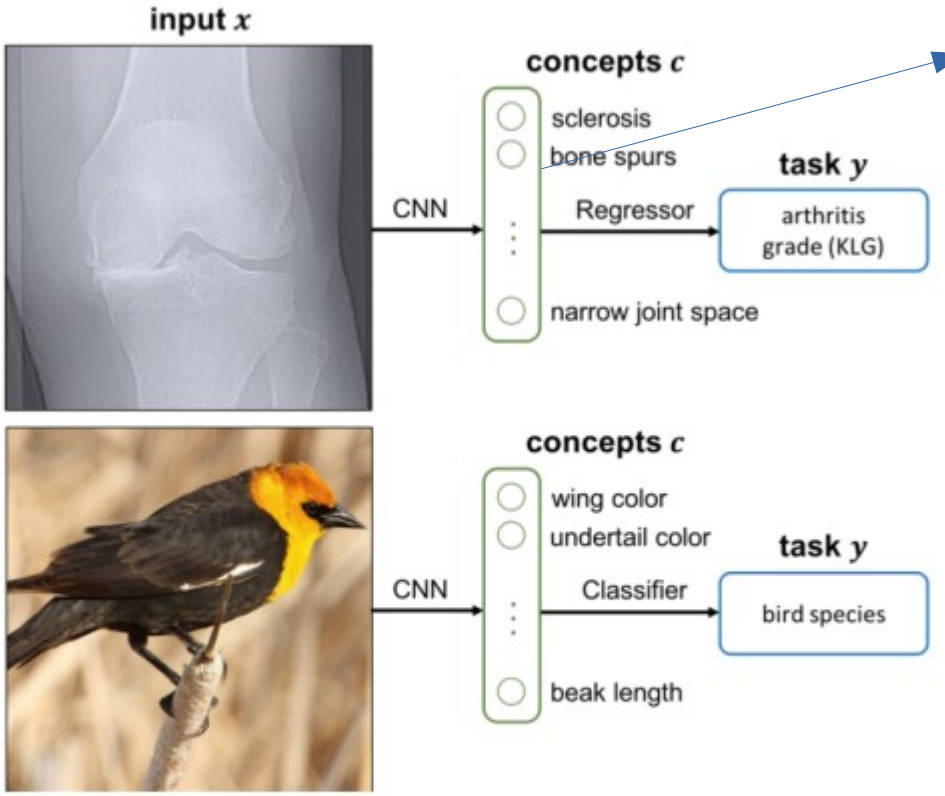


Figure 1. We study concept bottleneck models that first predict an intermediate set of human-specified concepts c, then use c to predict the final output y. We illustrate the two applications we consider: knee x-ray grading and bird identification.

Attach an additional layer with k neurons/concepts (bottleneck layer) to the penultimate layer of your model.

Let’s give an overview to the main idea

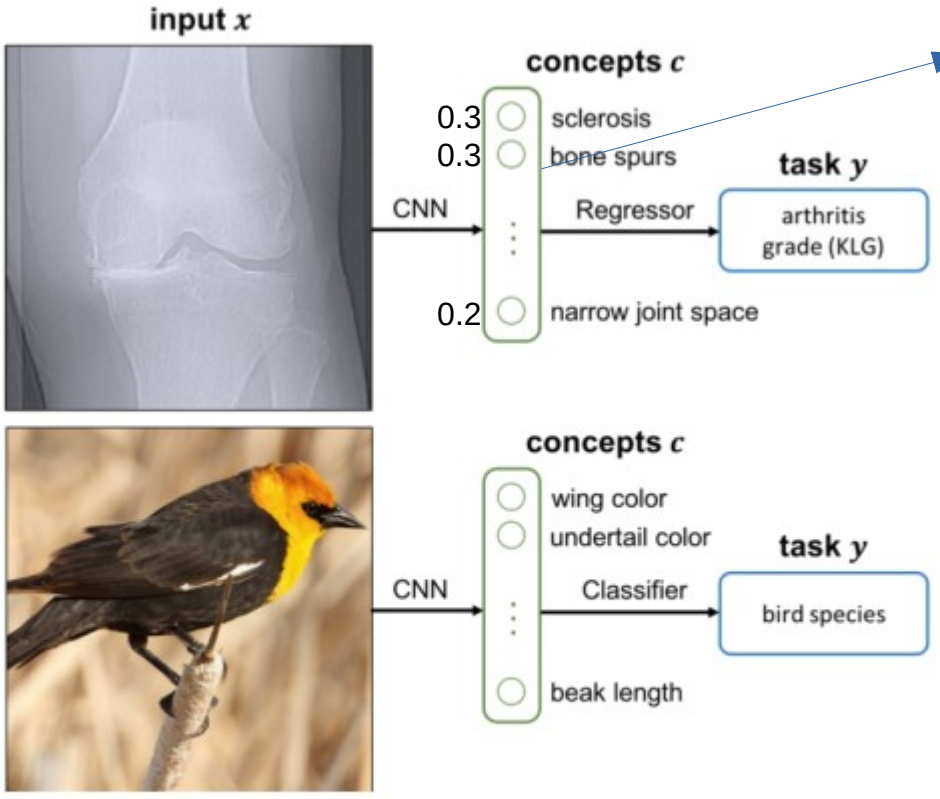


Figure 1. We study concept bottleneck models that first predict an intermediate set of human-specified concepts c, then use c to predict the final output y. We illustrate the two applications we consider: knee x-ray grading and bird identification.

Attach an additional layer with k neurons/concepts (bottleneck layer) to the penultimate layer of your model.

In the case of misclassification, the doctor can correct one or more concepts to obtain the correct result. Maybe the ‘narrow joint space’ concept should be higher!

Let’s give an overview to the main idea

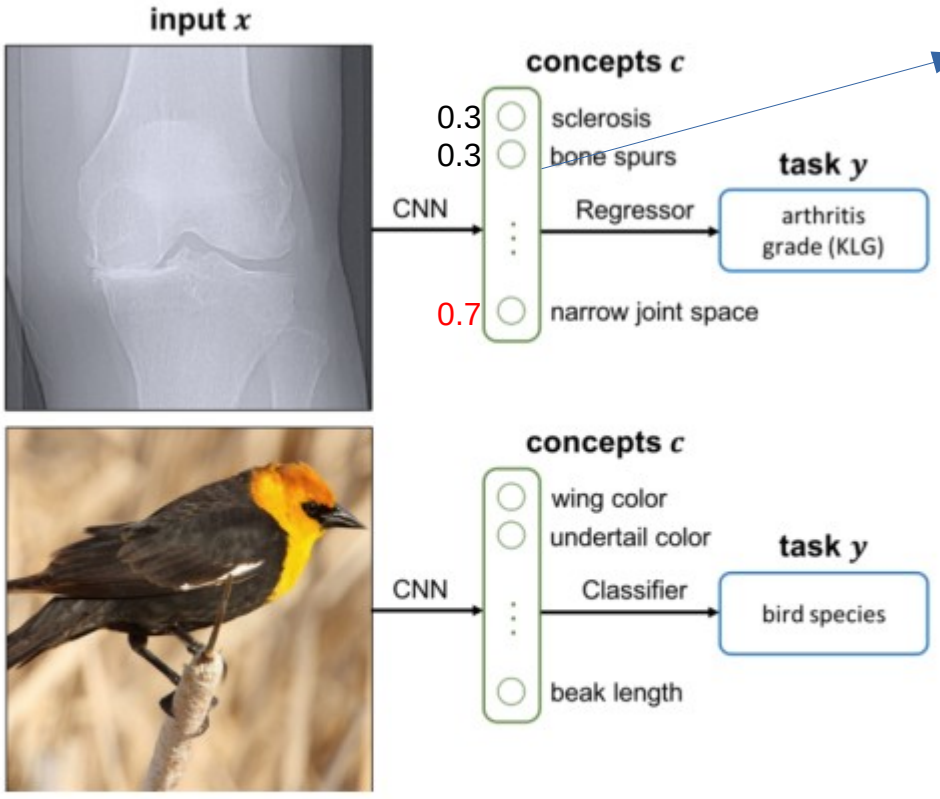


Figure 1. We study concept bottleneck models that first predict an intermediate set of human-specified concepts c, then use c to predict the final output y. We illustrate the two applications we consider: knee x-ray grading and bird identification.

Attach an additional layer with k neurons/concepts (bottleneck layer) to the penultimate layer of your model.

In the case of misclassification, the doctor can correct one or more concepts to obtain the correct result. Maybe the ‘narrow joint space’ concept should be higher!

Let’s give an overview to the main idea

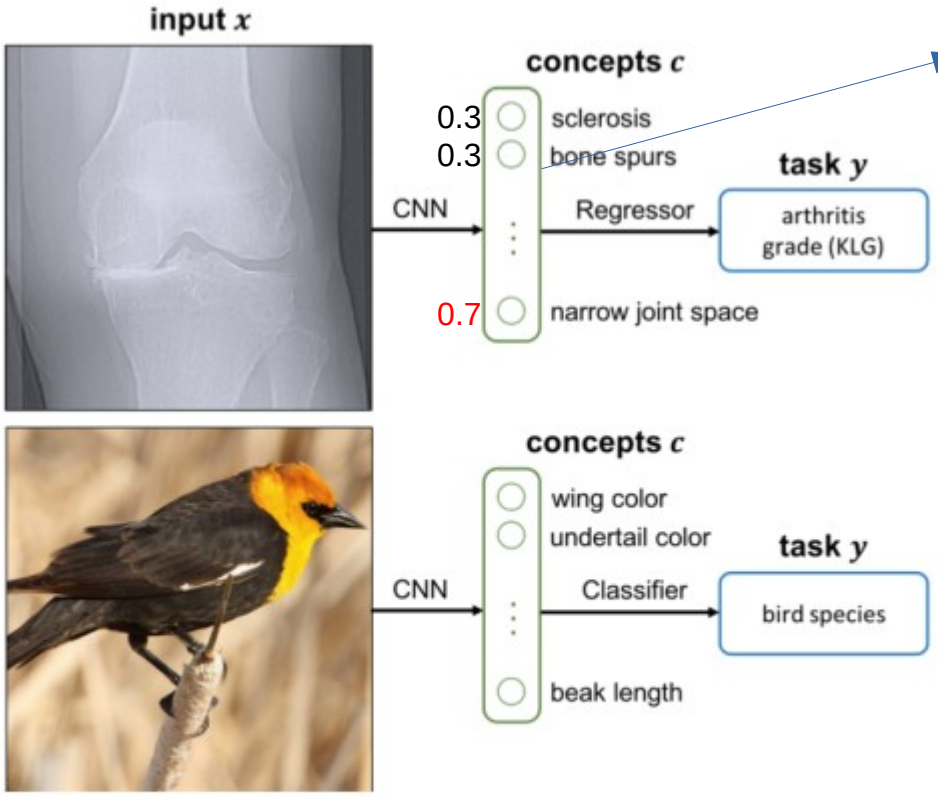


Figure 1. We study concept bottleneck models that first predict an intermediate set of human-specified concepts c, then use c to predict the final output y. We illustrate the two applications we consider: knee x-ray grading and bird identification.

Attach an additional layer with k neurons/concepts (bottleneck layer) to the penultimate layer of your model.

In the case of misclassification, the doctor can correct one or more concepts to obtain the correct result. Maybe the ‘narrow joint space’ concept should be higher!

Counterfactual explanations: What would happen to the prediction if you had a bone spur now?

Let’s give an overview to the main idea

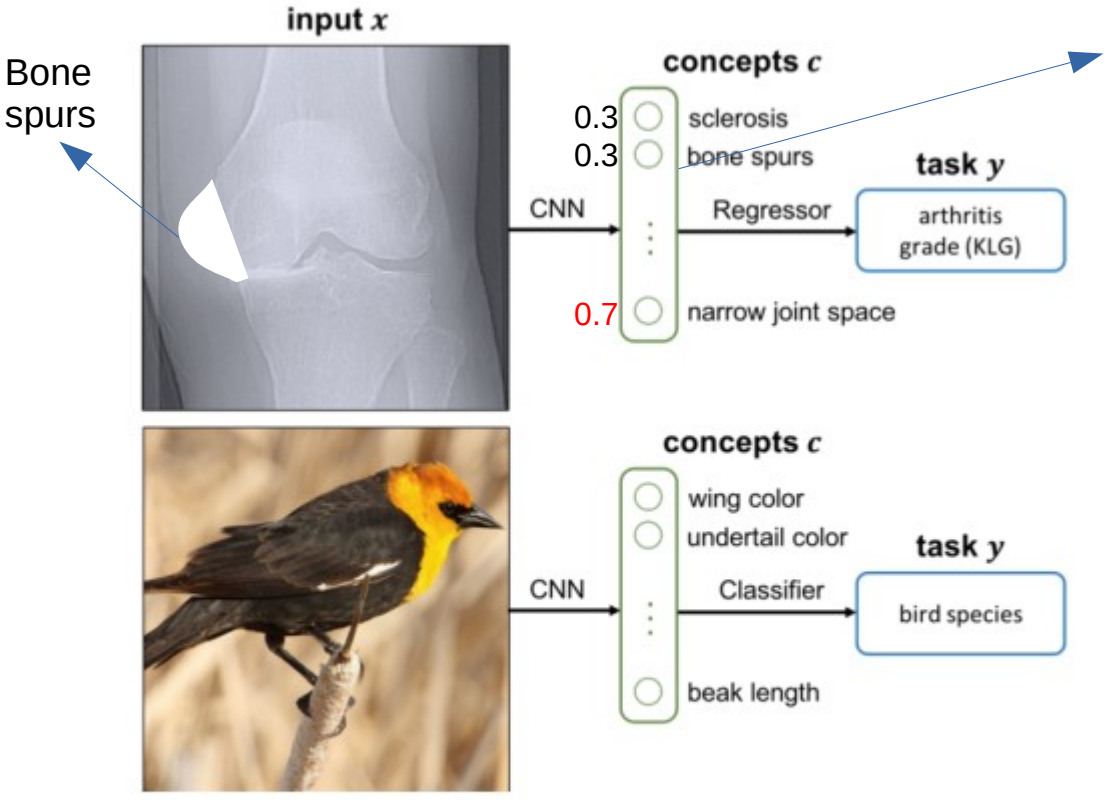


Figure 1. We study concept bottleneck models that first predict an intermediate set of human-specified concepts c, then use c to predict the final output y. We illustrate the two applications we consider: knee x-ray grading and bird identification.

Attach an additional layer with k neurons/concepts (bottleneck layer) to the penultimate layer of your model.

In the case of misclassification, the doctor can correct one or more concepts to obtain the correct result. Maybe the ‘narrow joint space’ concept should be higher!

Counterfactual explanations: What would happen to the prediction if you had a bone spur now?

Let’s give an overview to the main idea

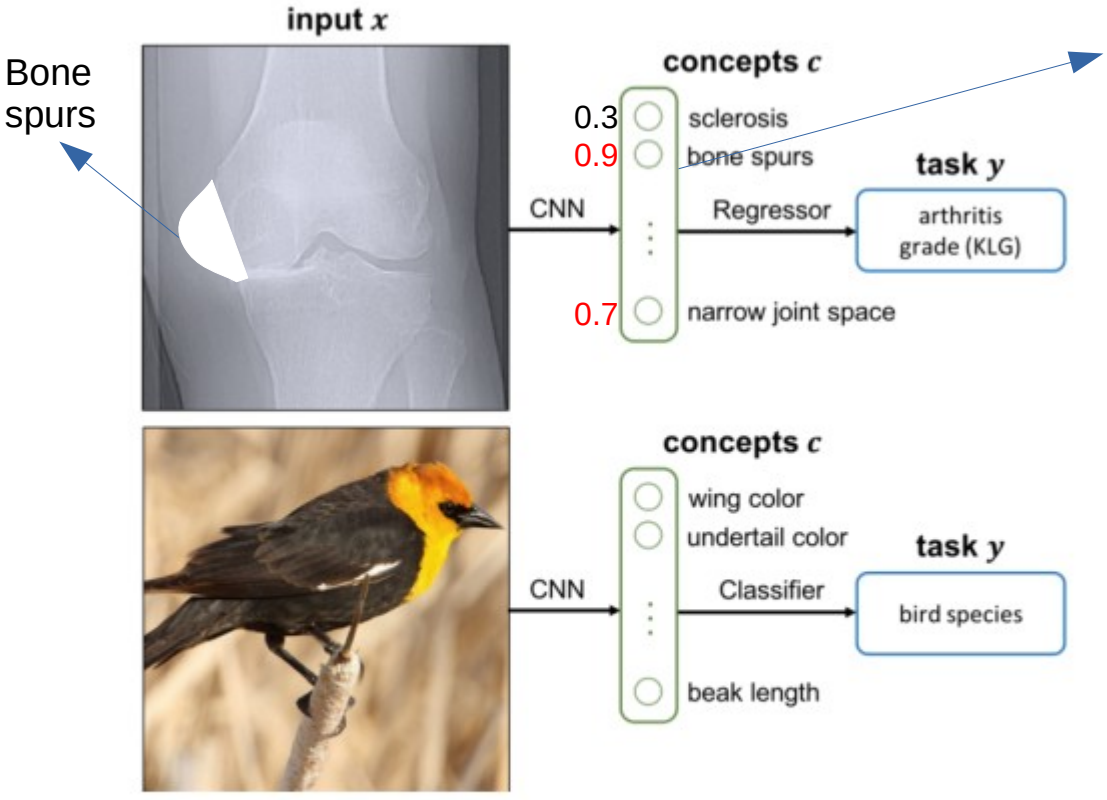


Figure 1. We study concept bottleneck models that first predict an intermediate set of human-specified concepts c, then use c to predict the final output y. We illustrate the two applications we consider: knee x-ray grading and bird identification.

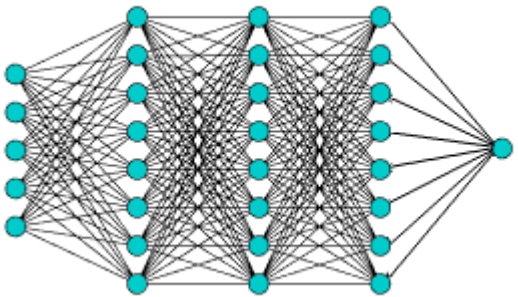
Attach an additional layer with k neurons/concepts (bottleneck layer) to the penultimate layer of your model.

In the case of misclassification, the doctor can correct one or more concepts to obtain the correct result. Maybe the ‘narrow joint space’ concept should be higher!

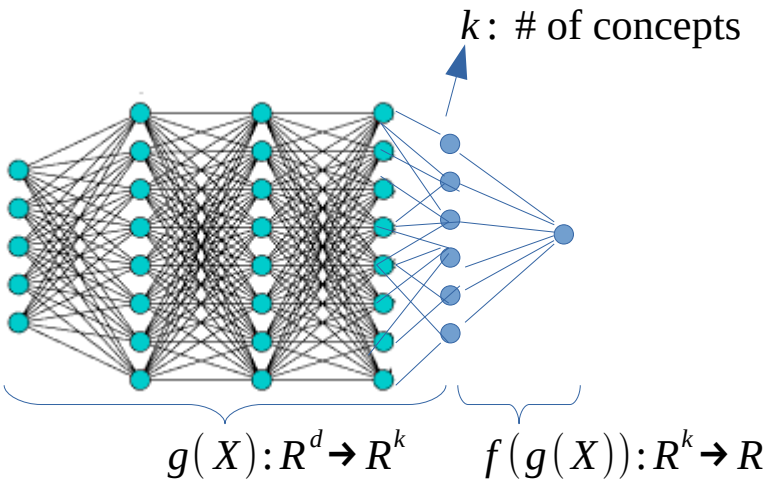
Counterfactual explanations: What would happen to the prediction if you had a bone spur now?

Implementation details

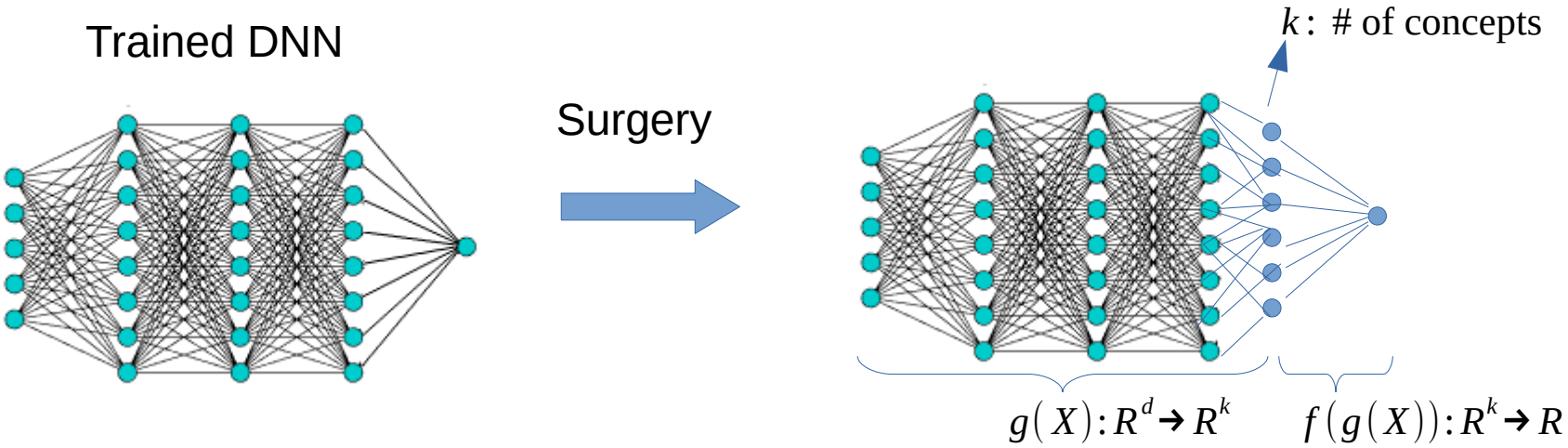
Trained DNN



Surgery

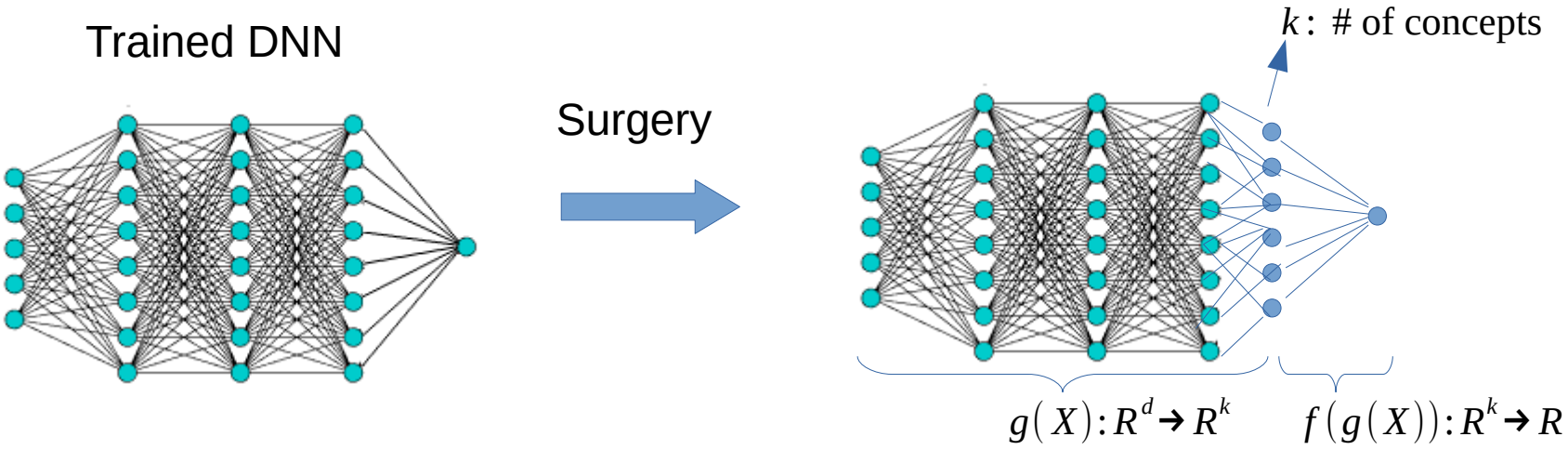


Implementation details



1 – Surgery: Attach the penultimate layer of your DNN to a new layer with as many neurons as the number of concepts you want to represent. Note that you will only need to train the last two layers (the blue connections)!

Implementation details

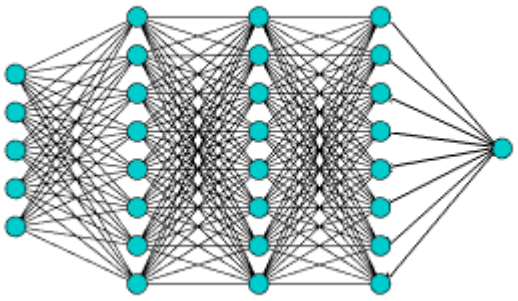


1 – Surgery: Attach the penultimate layer of your DNN to a new layer with as many neurons as the number of concepts you want to represent. Note that you will only need to train the last two layers (the blue connections)!

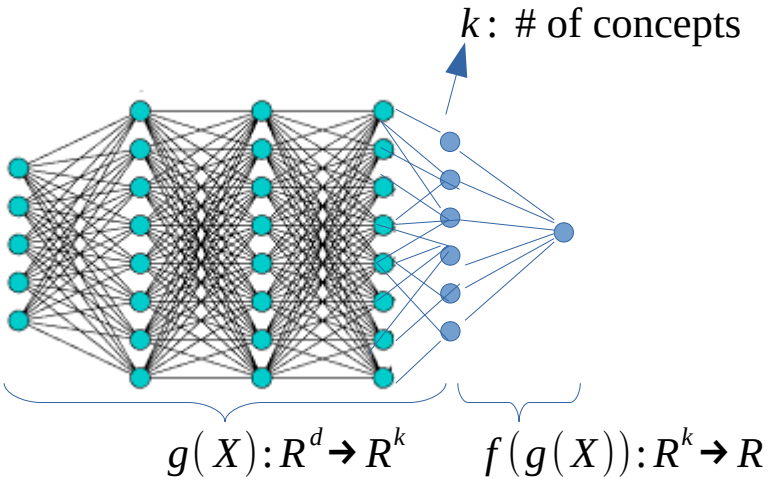
2 – f maps input to concepts and g maps concepts to labels.

Implementation details

Trained DNN



Surgery



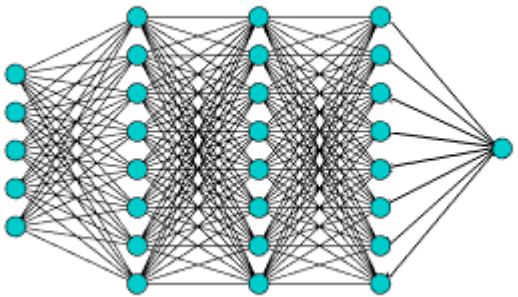
1 – Surgery: Attach the penultimate layer of your DNN to a new layer with as many neurons as the number of concepts you want to represent. Note that you will only need to train the last two layers (the blue connections)!

2 – f maps input to concepts and g maps concepts to labels.

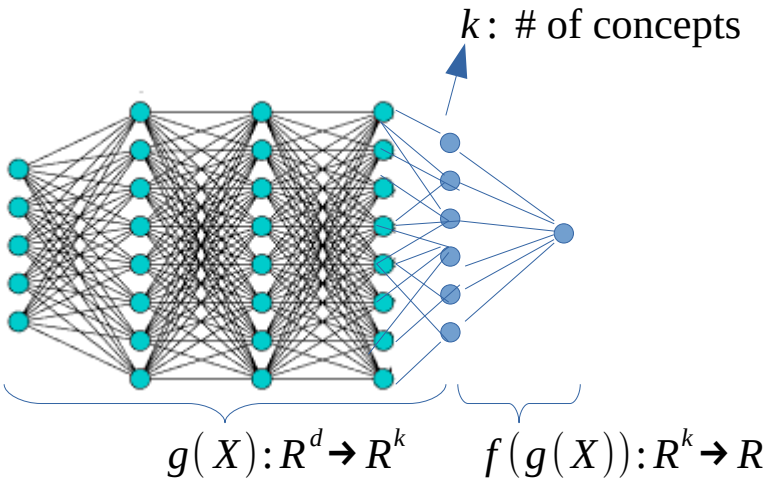
3 – To retrain the model we will need to have a dataset $\{(x^{(i)}, y^{(i)}, c^{(i)})\}_{i=1,..,n}$ with concepts $c^{(i)} \in R^k$.

Implementation details

Trained DNN



Surgery



1 – Surgery: Attach the penultimate layer of your DNN to a new layer with as many neurons as the number of concepts you want to represent. Note that you will only need to train the last two layers (the blue connections)!

2 – f maps input to concepts and g maps concepts to labels.

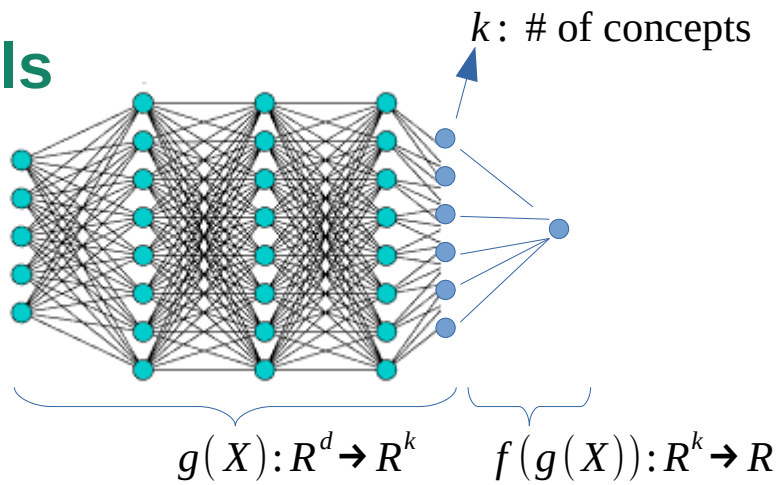
3 – To retrain the model we will need to have a dataset $\{(x^{(i)}, y^{(i)}, c^{(i)})\}_{i=1,..,n}$ with concepts $c^{(i)} \in R^k$.

4 – Design the cost functions for retraining the model.

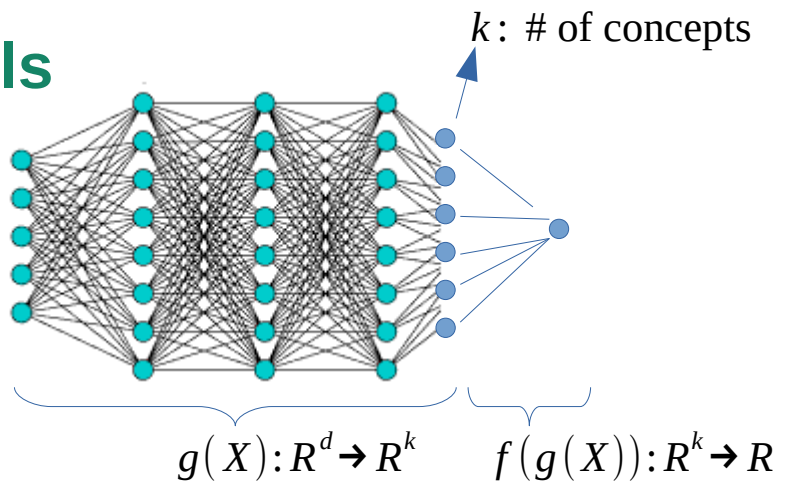
Implementation details

4 – Design the cost functions for retraining the model.

The paper proposes three approaches:



Implementation details



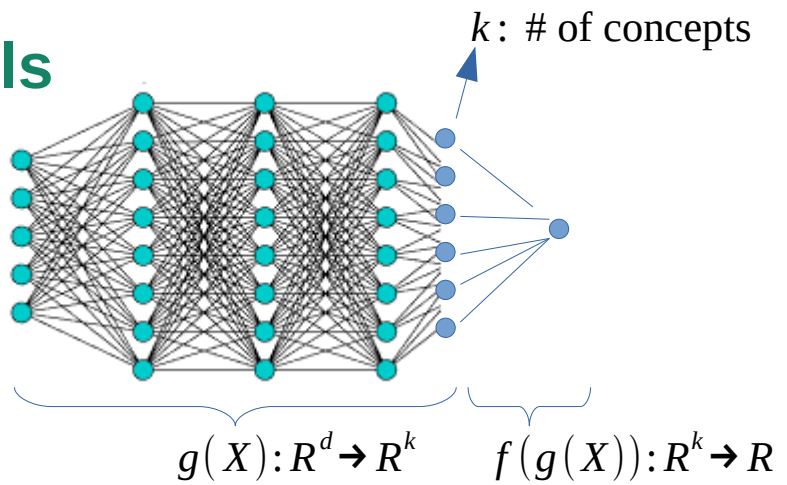
4 – Design the cost functions for retraining the model.

The paper proposes three approaches:

1. **The independent bottleneck** learns $\hat{g}$ and $\hat{f}$ independently:

$$\hat{g} = argmin_g \sum_{i,j} L_{C_j}(g_j(x^{(i)}), c_j^{(i)}) \qquad \hat{f} = argmin_f \sum_i L_Y(f(c^{(i)}), y^{(i)})$$

Implementation details



4 – Design the cost functions for retraining the model.

The paper proposes three approaches:

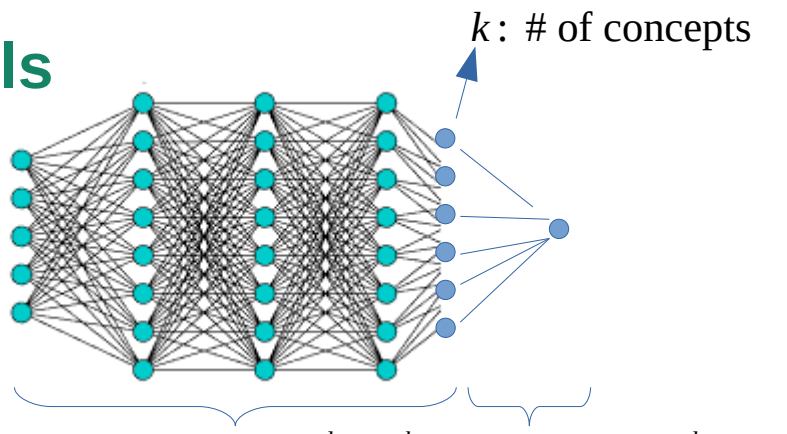
1. **The independent bottleneck** learns $\hat{g}$ and $\hat{f}$ independently:

$$\hat{g} = argmin_g \sum_{i,j} L_{C_j}(g_j(x^{(i)}), c_j^{(i)}) \qquad \hat{f} = argmin_f \sum_i L_Y(f(c^{(i)}), y^{(i)})$$

2. **The sequential bottleneck** first learns $\hat{g}$ in the same way as above. It then uses the concept predictions $\hat{g}$ to learn:

$$\hat{f} = argmin_f \sum_i L_Y(f(\hat{g}(x^{(i)})), y^{(i)})$$

Implementation details



4 – Design the cost functions for retraining the model.

The paper proposes three approaches:

1. **The independent bottleneck** learns $\hat{g}$ and $\hat{f}$ independently:

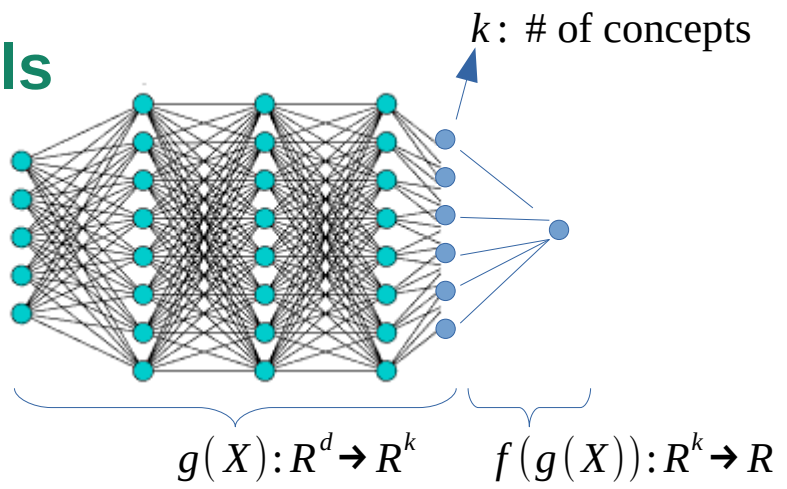
$$\hat{g} = argmin_g \sum_{i,j} L_{C_j}(g_j(x^{(i)}), c_j^{(i)}) \qquad \hat{f} = argmin_f \sum_i L_Y(f(c^{(i)}), y^{(i)})$$

2. **The sequential bottleneck** first learns $\hat{g}$ in the same way as above. It then uses the concept predictions $\hat{g}$ to learn:

$$\hat{f} = argmin_f \sum_i L_Y(f(\hat{g}(x^{(i)})), y^{(i)})$$

3. **Standard model** (useful for benchmarking): $f, \hat{g} = argmin_{f,g} \sum_i L_Y(f(g(x^{(i)}), y^{(i)}))$

Implementation details



4 – Design the cost functions for retraining the model.

The paper proposes three approaches:

1. **The independent bottleneck** learns $\hat{g}$ and $\hat{f}$ independently:

$$\hat{g} = argmin_g \sum_{i,j} L_{C_j}(g_j(x^{(i)}), c_j^{(i)}) \qquad \hat{f} = argmin_f \sum_i L_Y(f(c^{(i)}), y^{(i)})$$

2. **The sequential bottleneck** first learns $\hat{g}$ in the same way as above. It then uses the concept predictions $\hat{g}$ to learn:

$$\hat{f} = argmin_f \sum_i L_Y\left(f(\hat{g}(x^{(i)})), y^{(i)}\right)$$

3. **Standard model** (useful for benchmarking): $$f, \hat{g} = argmin_{f,g} \sum_i L_Y\left(f(g(x^{(i)}), y^{(i)})\right)$$

4. **The joint bottleneck** minimizes the weighted sum: $$(\hat{f}, \hat{g}) = argmin_{f,g} \sum_i L_Y\left(f(g(x^{(i)})), y^{(i)}\right) + \sum_j \lambda L_{C_j}\left(g(x^{(i)}), c_j^{(i)}\right)$$

Note that in point-3 the limit $\lambda \rightarrow \infty$ will produce the sequential bottleneck model and the limit $\lambda \rightarrow 0$ will recover the standard model.

Results

The paper works with two datasets:

X-ray grading, Osteoarthritis Initiative (OAI): Kellgren-Lawrence grade (KLG), a 4-level ordinal variable assessed by radiologists that measures the severity of osteoarthritis. Number of concepts, k=10. n~36K

Bird identification (CUB): The target are 200 different bird species. Number of concepts k=112 binary bird attributes like wing color, beak shape, etc.

Models used: Resnet-18 (OAI) and inception-v3 (CUB)

Results

Remember, Joint model:

$$(\hat{f},\hat{g}) = argmin_{f,g} \sum_i L_Y\left(f(g(x^{(i)})), y^{(i)}\right) + \sum_j \lambda L_{C_j}\left(g(x^{(i)}), c_j^{(i)}\right)$$

Target error

Model	y RMSE (OAI)	y Error (CUB)
Independent	0.435± 0.024	0.240±0.012
Sequential	0.418± 0.004	0.243±0.006
Joint	0.418± 0.004	0.199±0.006
Standard	0.441± 0.006	0.175±0.008
No bottleneck	0.443 ± 0.008	0.173±0.003
Multitask	0.425± 0.010	0.162±0.002

From the three proposed models, the Joint, is the best one!

Concept error

Model	c RMSE (OAI)	c Error (CUB)
Independent	0.529±0.004	0.034±0.002
Sequential	0.527±0.004	0.034±0.002
Joint	0.543±0.014	0.031±0.000
Standard [probe]	0.680±0.038	0.093±0.004
SENN [probe]	0.676±0.026	-

Results

Remember, Joint model:

$$(\hat{f},\hat{g}) = argmin_{f,g} \sum_i L_Y\left(f(g(x^{(i)})), y^{(i)}\right) + \sum_j \lambda L_{C_j}\left(g(x^{(i)}), c_j^{(i)}\right)$$

Target error

Model	y RMSE (OAI)	y Error (CUB)
Independent	0.435± 0.024	0.240±0.012
Sequential	0.418± 0.004	0.243±0.006
Joint	0.418± 0.004	0.199±0.006
Standard	0.441± 0.006	0.175±0.008
no bottleneck	0.443 ± 0.008	0.173±0.003
Multitask	0.425± 0.010	0.162±0.002

From the three proposed models, the Joint, is the best one!

RMSE for the OAI dataset is better for the three models compared to the non-bottleneck model. I suspect this is because the concepts are highly correlated with the target.

Concept error

Model	c RMSE (OAI)	c Error (CUB)
Independent	0.529±0.004	0.034±0.002
Sequential	0.527±0.004	0.034±0.002
Joint	0.543±0.014	0.031±0.000
Standard [probe]	0.680±0.038	0.093±0.004
SENN [probe]	0.676±0.026	-

Results

Remember, Joint model:

$$(\hat{f},\hat{g}) = argmin_{f,g} \sum_i L_Y\left(f(g(x^{(i)})), y^{(i)}\right) + \sum_j \lambda L_{C_j}\left(g(x^{(i)}), c_j^{(i)}\right)$$

Target error

Model	y RMSE (OAI)	y Error (CUB)
Independent	0.435± 0.024	0.240±0.012
Sequential	0.418± 0.004	0.243±0.006
Joint	0.418± 0.004	0.199±0.006
Standard	0.441± 0.006	0.175±0.008
no bottleneck	0.443 ± 0.008	0.173±0.003
Multitask	0.425± 0.010	0.162±0.002

From the three proposed models, the Joint, is the best one!

RMSE for the OAI dataset is better for the three models compared to the non-bottleneck model. I suspect this is because the concepts are highly correlated with the target.

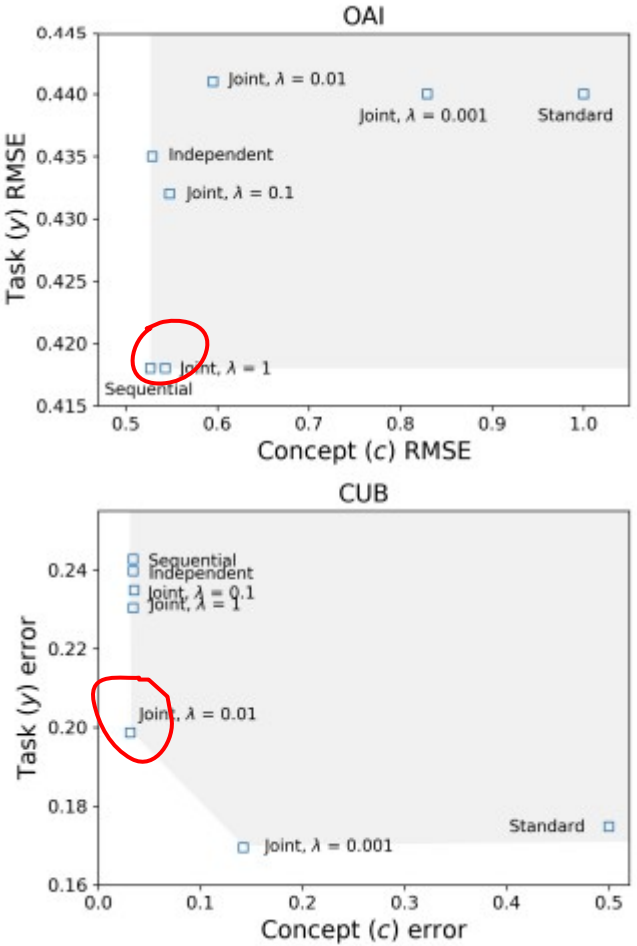
Concept error

Model	c RMSE (OAI)	c Error (CUB)
Independent	0.529±0.004	0.034±0.002
Sequential	0.527±0.004	0.034±0.002
Joint	0.543±0.014	0.031±0.000
Standard [probe]	0.680±0.038	0.093±0.004
SENN [probe]	0.676±0.026	-

The accuracy on CUB is quite high ~ 97% .

This should be the case in practice; otherwise, relying on concepts wouldn't be beneficial.

Results

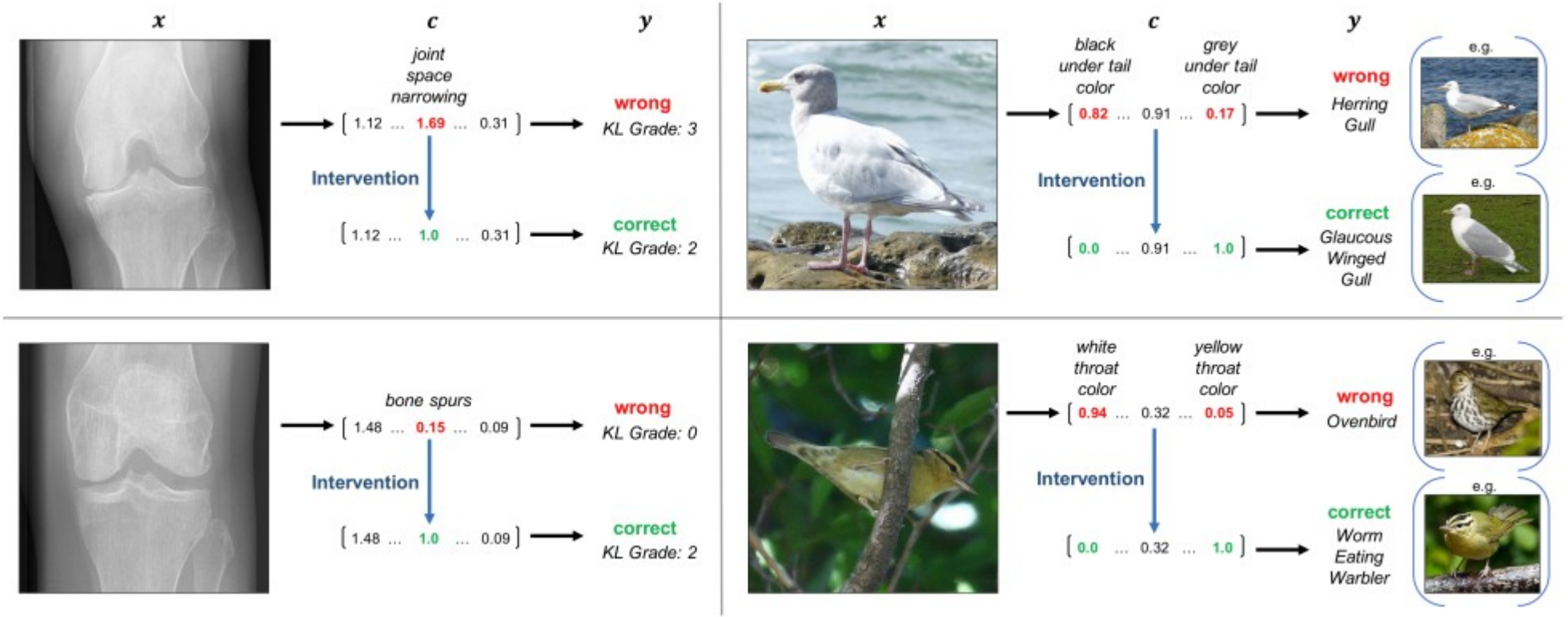


The points in red are the ones we are interested in, as we aim for the highest possible concept accuracy while maintaining good task accuracy.

In general, the joint model is performing better, as expected!

Results

Concept Intervention



General questions:

1 - If the bottleneck is not very representative, could the model's performance decrease?

2 - In the case of osteoarthritis, where concepts are highly correlated with the target, does the model find it easier to understand them?

3 - When conducting concept intervention with 100 concepts, how should you proceed? How do you determine which concepts to adjust, and what are the appropriate values?

Post-hoc CBM (PCBM)

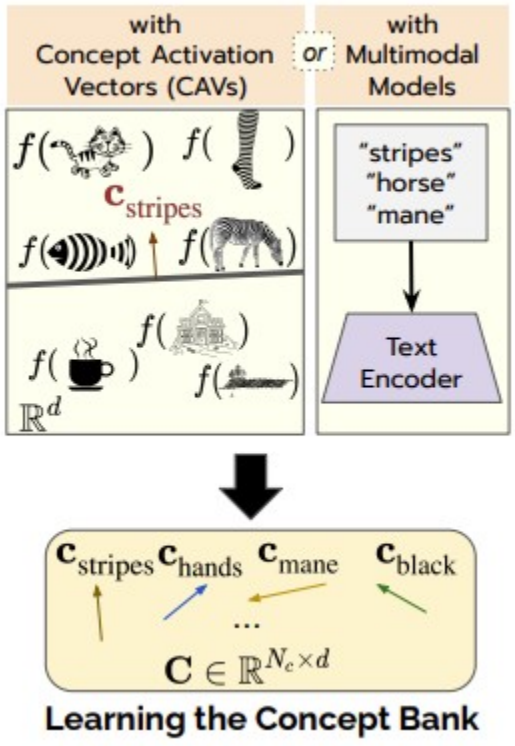
Combining the strengths of TCAV and CBM, it offers an easy way to create concepts similar to TCAV (even simpler) while providing the ability to interact with them as in CBM.

Yuksekgonul et. al. - Post-hoc Concept Bottleneck Models - ICLR 2023

PCBM idea

CAV generation:

A – TCAV approach

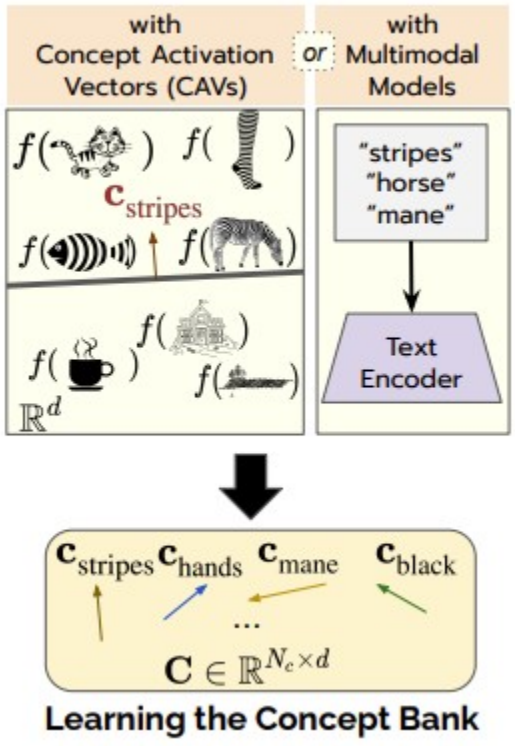


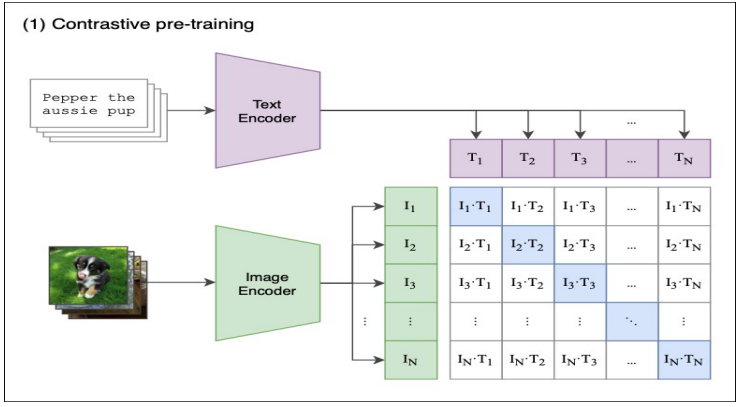
PCBM idea

CAV generation:

A – TCAV approach

B – Multimodal Models (Image/Test like CLIP)



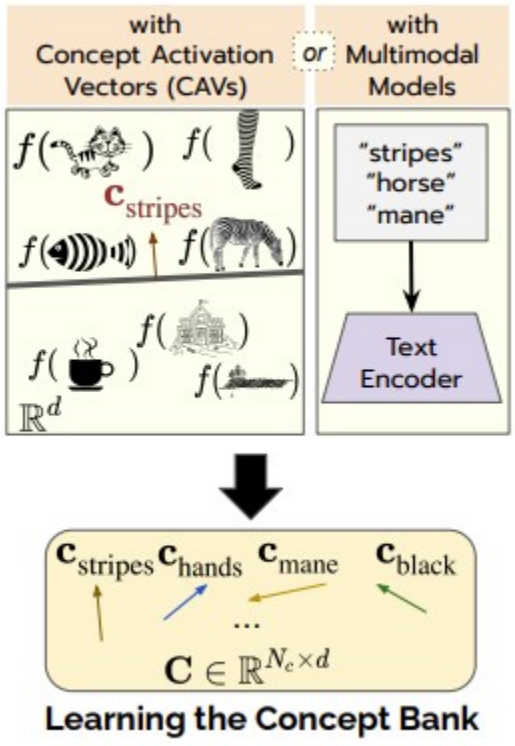


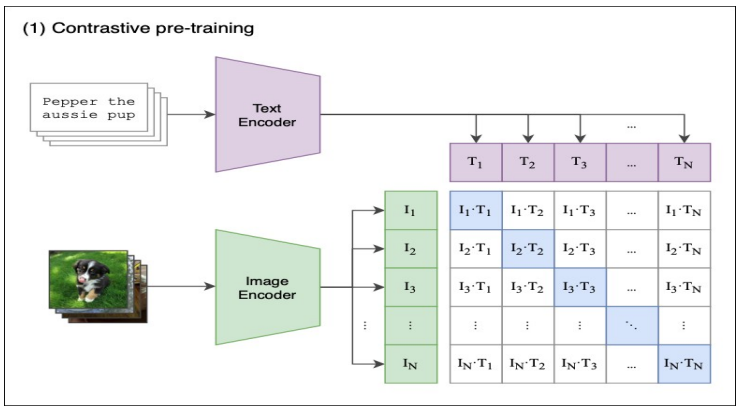
PCBM idea

CAV generation:

A – TCAV approach

B – Multimodal Models (Image/Test like CLIP)



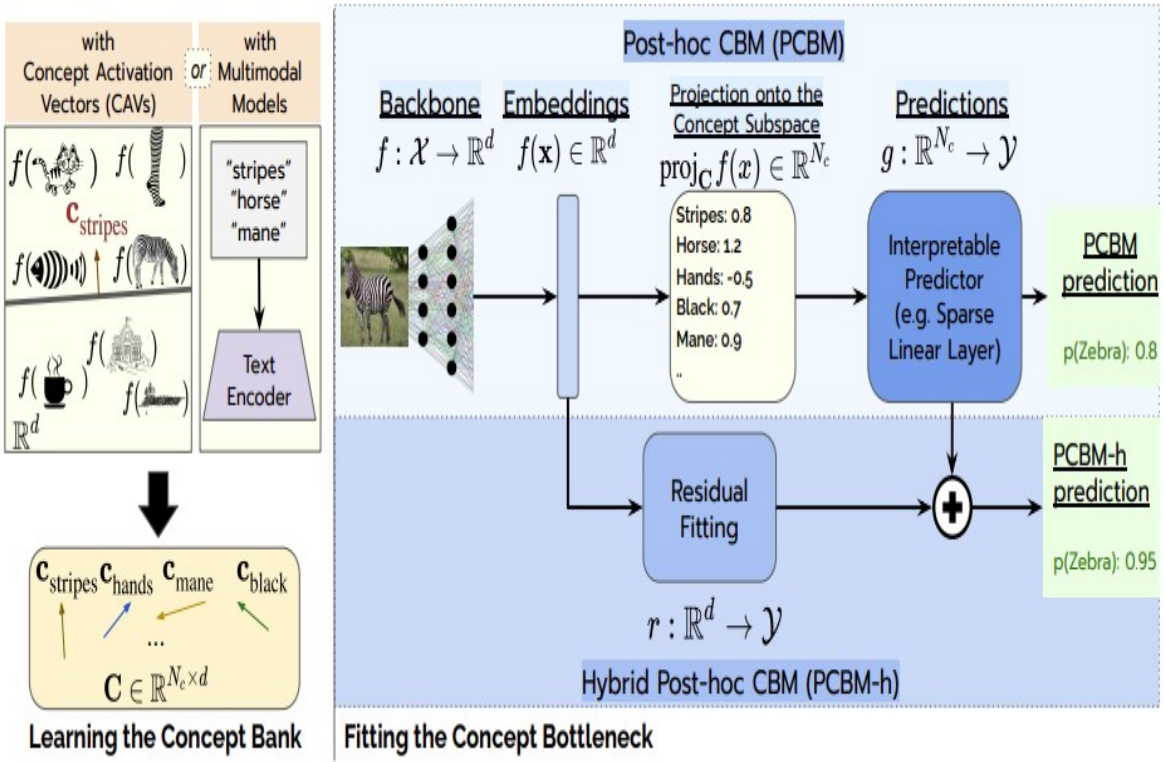


Q. We cannot expect TCAV and CLIP embedding to be similar, right?. However there is a retraining process …

PCBM idea

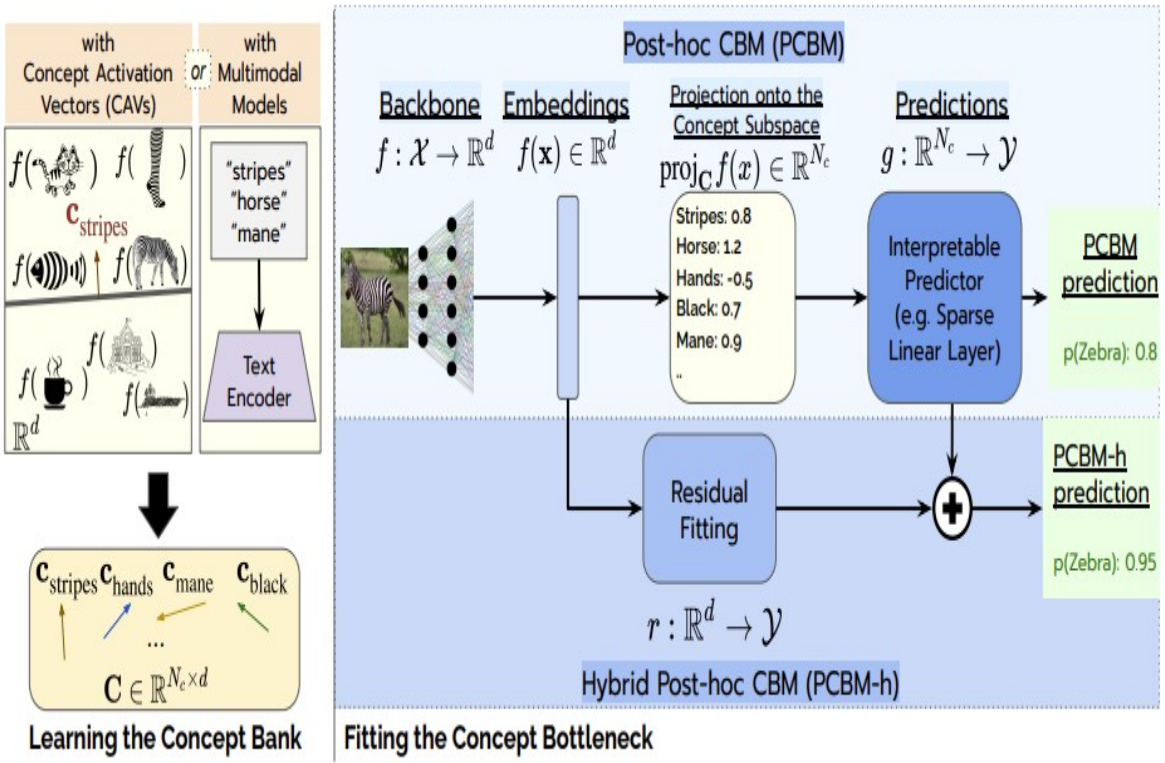
PCBM architecture:

1 – **Learning the concept bank**



PCBM idea

PCBM architecture:

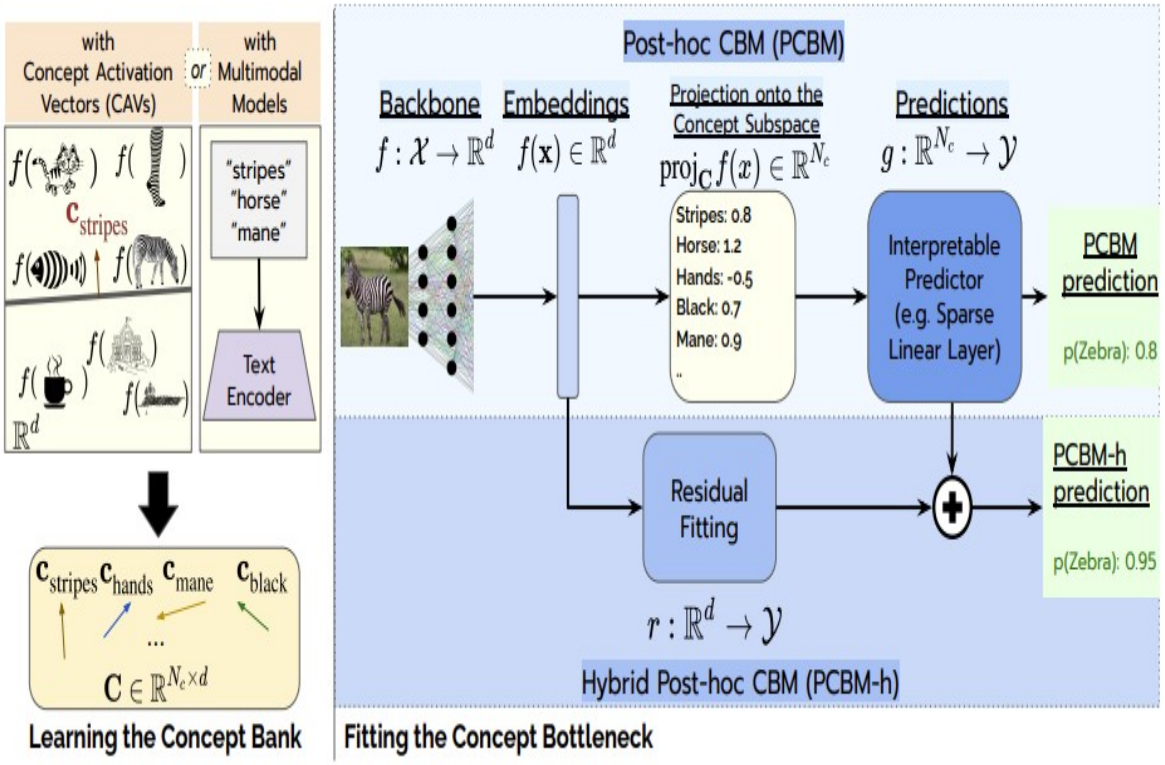


1 – **Learning the concept bank**

2 – **Backbone model**: This is your trained DNN with the penultimate layer attached to the embedding $f(X)$ (similar to what we did in CBM).

PCBM idea

PCBM architecture:



1 – **Learning the concept bank**

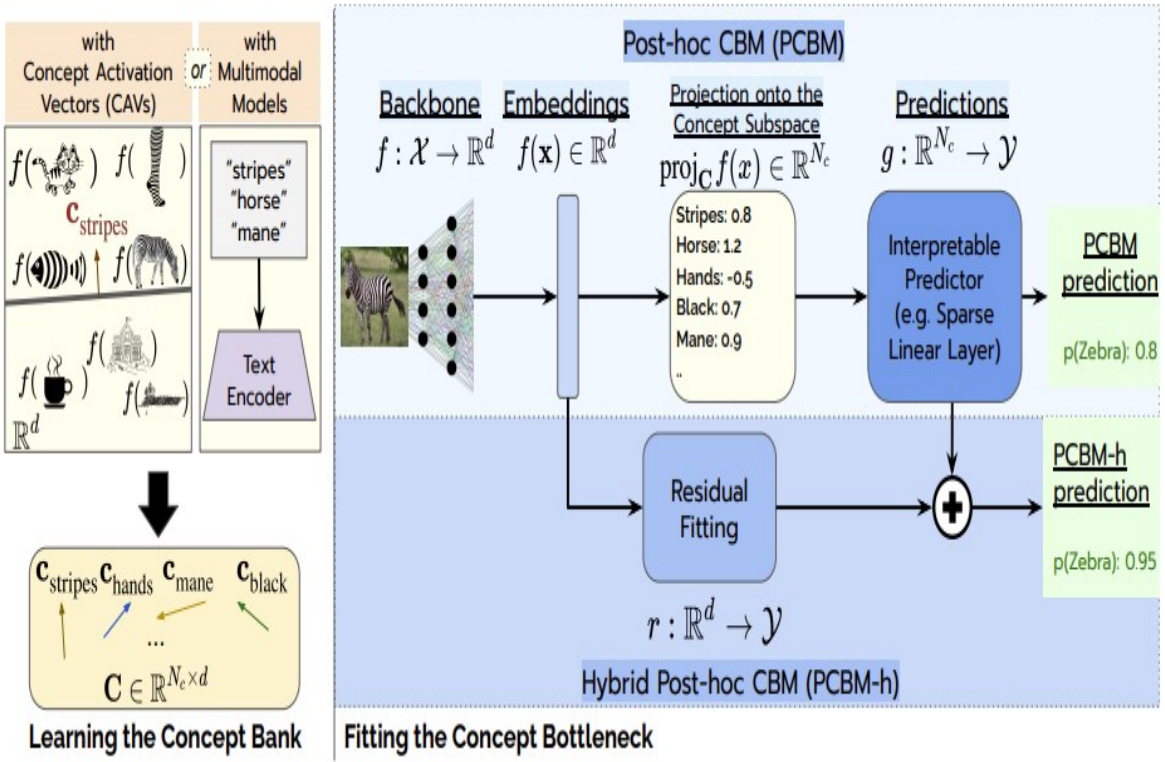
2 – **Backbone model**: This is your trained DNN with the penultimate layer attached to the embedding $f(X)$ (similar to what we did in CBM).

3 – **Projection into concept subspace.**

$$\text{proj}_C f^{(i)}(X) = \frac{\langle f(x), c_i \rangle}{\|c_i\|^2} \quad \text{with } i = 1, .., N_C$$

PCBM idea

PCBM architecture:



1 – **Learning the concept bank**

2 – **Backbone model**: This is your trained DNN with the penultimate layer attached to the embedding $f(X)$ (similar to what we did in CBM).

3 – **Projection into concept subspace.**

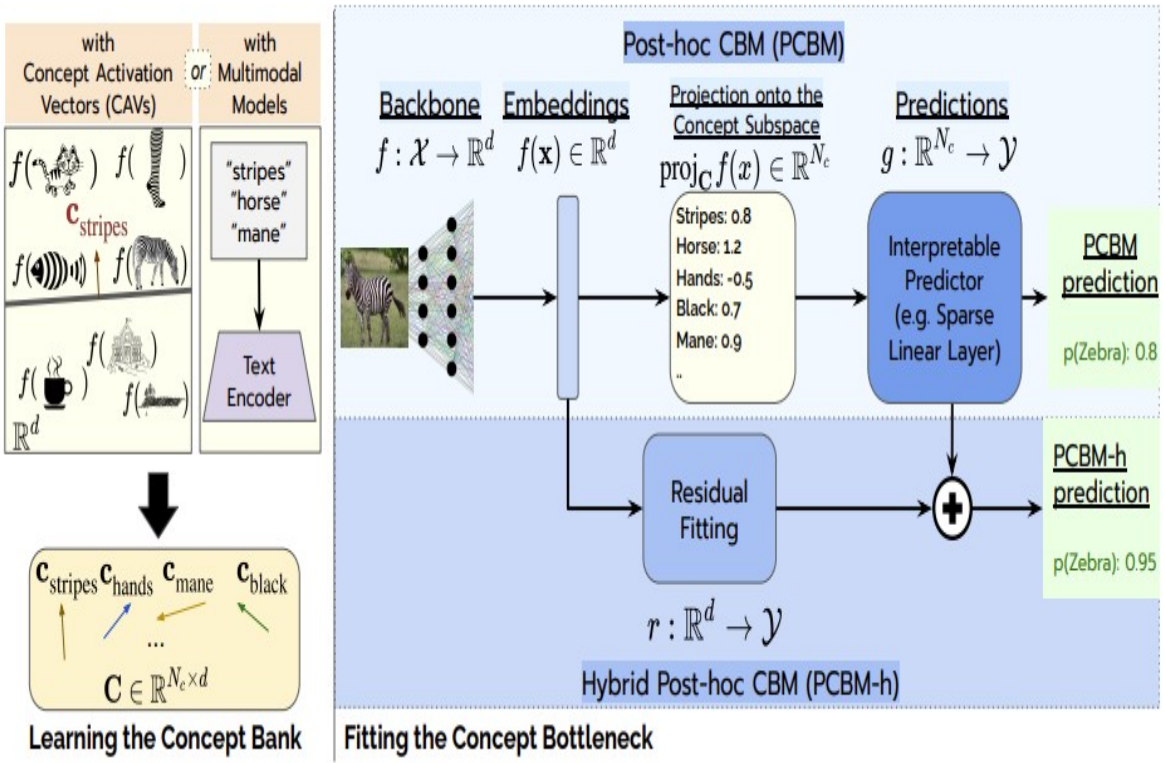
$$\text{proj}_C f^{(i)}(X) = \frac{\langle f(x), c_i \rangle}{\|c_i\|^2} \quad \text{with } i = 1, .., N_C$$

4 – **Prediction (to be trained)**

$$g : R^{N_C} \to y$$

PCBM idea

PCBM architecture:



1 – **Learning the concept bank**

2 – **Backbone model**: This is your trained DNN with the penultimate layer attached to the embedding $f(X)$ (similar to what we did in CBM).

3 – **Projection into concept subspace.**

$$\text{proj}_C f^{(i)}(X) = \frac{\langle f(x), c_i \rangle}{\|c_i\|^2} \quad \text{with } i = 1, .., N_C$$

4 – **Prediction (to be trained)**

$$g : R^{N_C} \to y$$

5 – **Residual Fitting**: If prediction space is not representative enough we could have a decrease in original's model performance.

Residual Fitting Module: focuses in accuracy rather than concepts.

The model with the residual module is call hybrid PCBM (PCBMh)

Experiments

PCBMs achieve comparable performance to the original model

	CIFAR10	CIFAR100	COCO-Stuff	CUB	HAM10000	ISIC
Original Model	0.888	0.701	0.770	0.612	0.963	0.821
PCBM	0.777 ± 0.003	0.520 ± 0.005	0.741 ± 0.002	0.588 ± 0.008	0.947 ± 0.001	0.736 ± 0.012
PCBM-h	0.871 ± 0.001	0.680 ± 0.001	0.768 ± 0.01	0.610 ± 0.010	0.962 ± 0.002	0.801 ± 0.056

Observations:

PCBMs match the performance of the original model in HAM10000 and ISIC, with as few as 8 human-interpretable concepts

In CIFAR100, it seems that the concept bank available is not sufficient to classify finer-grained classes

The residual component of PCBMh only intervenes when the prediction is wrong and fixes the mistake. In general this happened when the concept bank is not sufficiently expressive.

Experiments

PCBM using CLIP concepts:

	CIFAR10	CIFAR100	COCO-Stuff
Original Model (CLIP)	0.888	0.701	0.770
PCBM & labeled concepts	0.777 ± 0.003	0.520 ± 0.005	0.741 ± 0.002
PCBM & CLIP concepts	0.833 ± 0.003	0.600 ± 0.003	0.755 ± 0.001
PCBM-h & CLIP concepts	0.874 ± 0.001	0.691 ± 0.006	0.769 ± 0.001
# of CLIP concepts:	206	527	822

Observations:

The original model is CLIP so $f(X)$ resides in the same encoding space as the CAVs. It will be nice to see what happens with a different ‘Original Model’.

If the concept space is too large how do you interact with it?

Experiments

Human concept intervention

Conclusions

- The notion of high-level concepts is appealing and can be beneficial in real applications.

- A method like PCBM, which combines the strengths of both TCAV and CBM, and offers the flexibility to incorporate CAVs from multimodal models, has the potential to be a game-changer for concept-based interpretable Neural Networks.

- However, it still appears impractical because the concept spaces are too large. Nonetheless, as demonstrated in the PCBM paper, human intervention can help reduce the concept space while maintaining accuracy but the evidence is not conclusive to be used in practice.

- Another issue is that multimodal models may struggle to generate accurate CAVs in highly specialized fields.

Thank you !!!