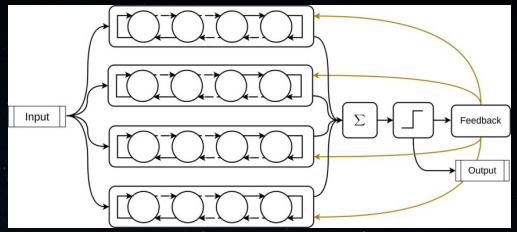
Tsetlin Machines



A very brief introduction

frank.kelly@cantab.net

frankk@sahaj.ai

Why use Tsetlin Machines?

+ Highly accurate algorithms
+ Powerful computational platforms
- Escalating computational costs
- Ever more complicated models

layer name	output size	18-layer	34-layer	50-layer	101-layer	152-layer
conv1	112×112	7×7, 64, stride 2				
		3×3 max pool, stride 2				
conv2_x	56×56	[3×3, 64; 3×3, 64] ×2	[3×3, 64; 3×3, 64] ×3	[1×1, 64; 3×3, 64; 1×1, 256] ×3	[1×1, 64; 3×3, 64; 1×1, 256] ×3	[1×1, 64; 3×3, 64; 1×1, 256] ×3
conv3_x	28×28	[3×3, 128; 3×3, 128] ×2	[3×3, 128; 3×3, 128] ×4	[1×1, 128; 3×3, 128; 1×1, 512] ×4	[1×1, 128; 3×3, 128; 1×1, 512] ×4	[1×1, 128; 3×3, 128; 1×1, 512] ×8
conv4_x	14×14	[3×3, 256; 3×3, 256] ×2	[3×3, 256; 3×3, 256] ×6	[1×1, 256; 3×3, 256; 1×1, 1024] ×6	[1×1, 256; 3×3, 256; 1×1, 1024] ×23	[1×1, 256; 3×3, 256; 1×1, 1024] ×36
conv5_x	7×7	[3×3, 512; 3×3, 512] ×2	[3×3, 512; 3×3, 512] ×3	[1×1, 512; 3×3, 512; 1×1, 2048] ×3	[1×1, 512; 3×3, 512; 1×1, 2048] ×3	[1×1, 512; 3×3, 512; 1×1, 2048] ×3
	1×1	average pool, 1000-d fc, softmax				
FLOPs		1.8×10^9	3.6×10^9	3.8×10^9	7.6×10^9	11.3×10^9

ures for ImageNet. Building blocks are shown in brackets (see also Fig. 5), with the numbers of block

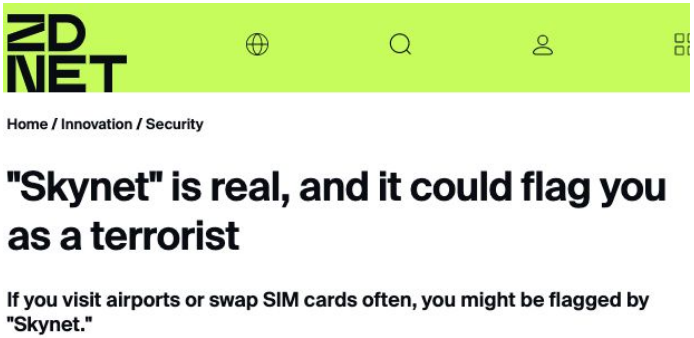


Image from: https://pytorch.org/hub/pytorch_vision_resnet/

Tsetlin machines are:

- Universal function approximators (like NNs)
- Rule-based (like decision trees)
- Summation-based (like naïve Bayes)
- Low energy and memory footprint (hardware-near)

A general-purpose, interpretable, and low-energy machine learning approach

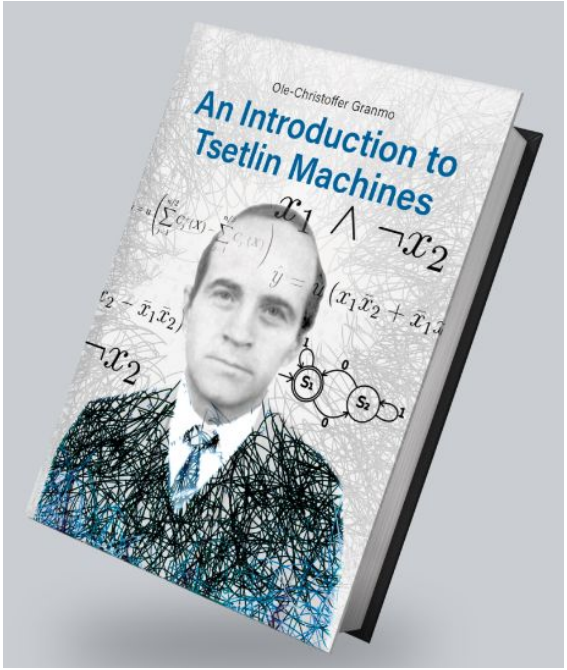
(apparently)

Is it possible?

How do Tsetlin machines work?

Picked up this book.

My “mini” research prior to engagement with a client project



https://tsetlinmachine.org/

1. Data Booleanisation

By making suitable boolean features

Turn inputs into **boolean** features

- Computers can understand it (IF, ELSE, AND etc.)
- Humans get it too (TRUE or FALSE statements)
- Use the features and logical operators to make a clause

2. Build Clauses (rules)

Boolean literals with operators, combined

Randomly select a set of features from the input:

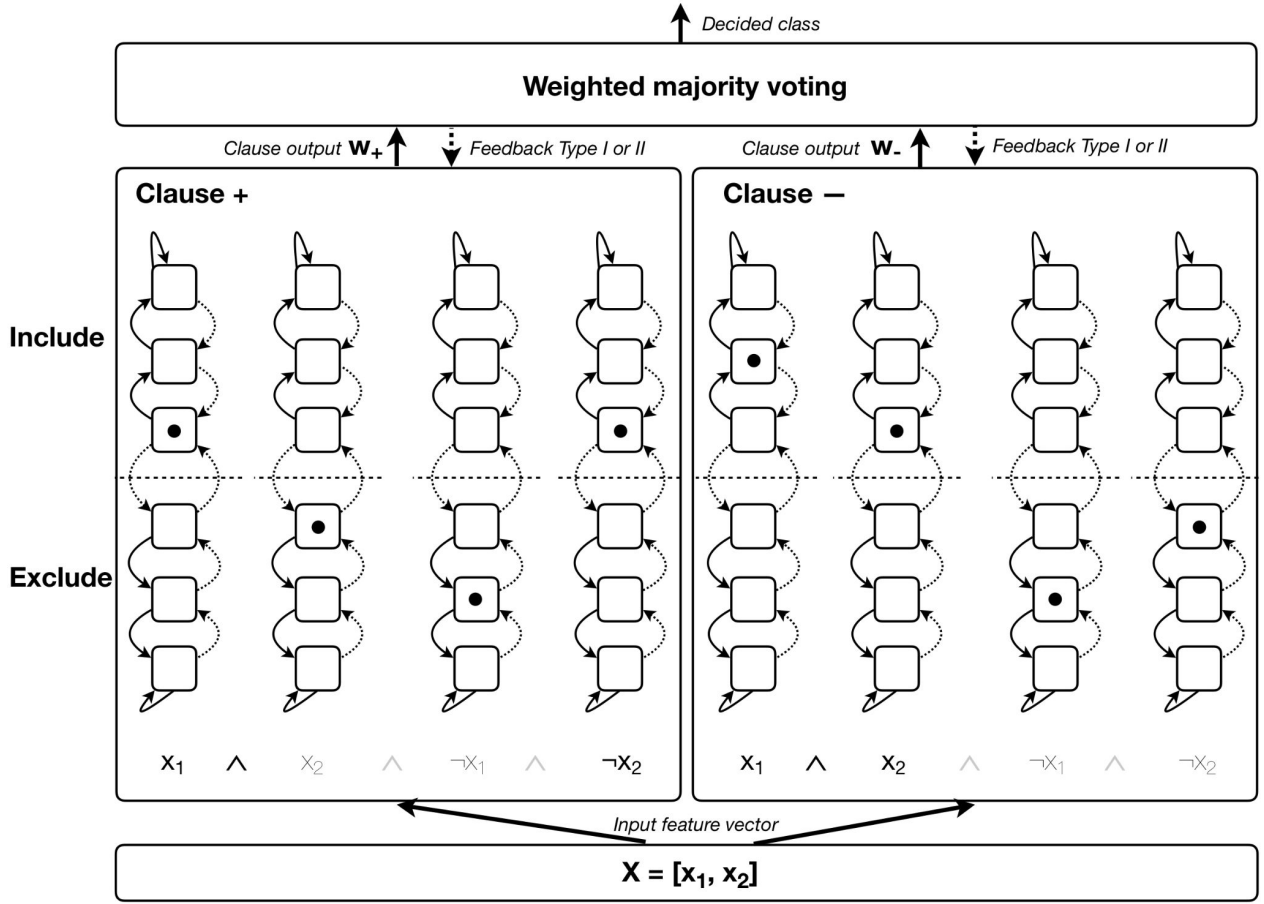
- Build IF-THEN **rules** based on observations (e.g. **If** *Four Wheels* **and** *Transports People* THEN *Car*)
- Include **literals** (features and negated features together)
- Each rule belongs to a **class** of objects, and learns to recognise objects of that class

3. Count, vote and co-ordinate

Include the best clauses and exclude the rest

The Tsetlin training process:

- Count the number of times a clause is satisfied by a given input
- A majority vote among the rules decides the output
- The uniqueness of the pattern per class is also important
- Coordinate with data dissection using a vote margin



https://github.com/olegranmo/blob/blob/master/Weighted_Tsetlin_Machine_Example_Configuration_Full.png

What data types can it handle?

* sequential
* temporal
* time series
* text

* images
* videos
* audio

I tried it out with simple regression ->

```
def __create_tsetlin_machine_regression_model() -> RegressionTsetlinMachin
    """
    Create a Tsetlin Machine regression model for time series prediction.
    """
    logging.info("Creating Tsetlin Machine regression model...")
    tm = RegressionTsetlinMachine(
        __number_of_clauses,
        __s,
        __number_of_state_bits,
        number_of_targets=1,
        weighted_clauses=True,
    )
    return tm


def __train_tsetlin_machine_regression_model(t_model, X_train, y_train):
    """
    Train a Tsetlin Machine regression model.
    """
    t_model.fit(X_train, y_train, epochs=__number_of_epochs)
    return t_model


def __predict_with_tsetlin_machine_regression_model(
    model: Model, data: Dataset, X_test: pd.DataFrame
) -> PredictionData:
    """
    Predict with a Tsetlin Machine regression model.
    """
    title = f"{data.subset_column_name} forecast for {data.subset_row_name
    return PredictionData(
        values=model.predict(X_test),
        prediction_column_name=None,
        ground_truth_values=X_test,
        confidence_columns=None,
```

Conclusion

Warrants further exploration!

Potential large-scale impact in:

- AI and ethics?
- Computing and climate change?

"...Yet simplicity has been difficult to implement in modern life because it is against the spirit of a certain brand of people who seek sophistication so they can justify their profession."

Nassim N. Taleb

Find out more

https://tsetlinmachine.org/

https://github.com/cair/pyTsetlinMachine

frank.kelly@cantab.net

frankk@sahaj.ai