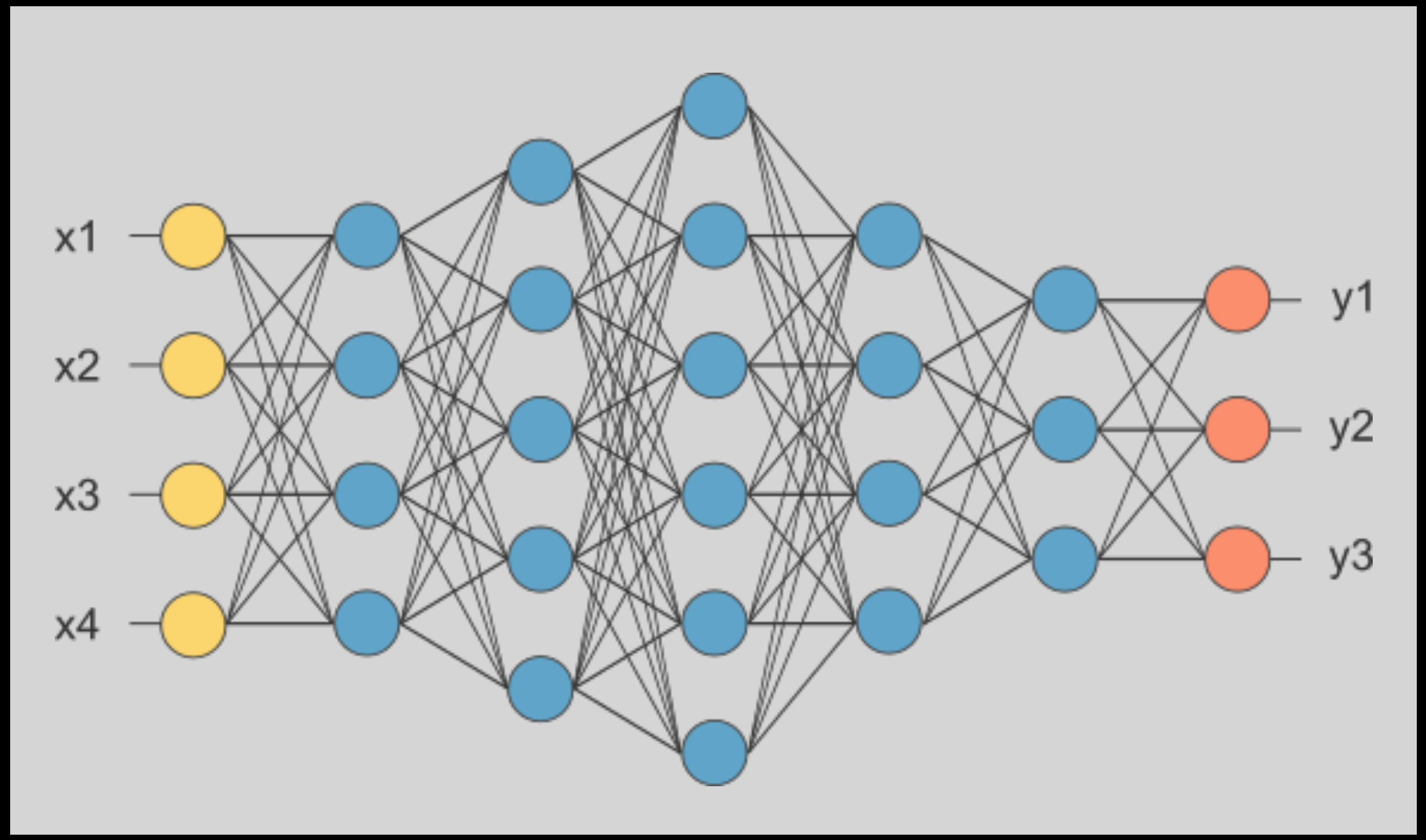


Reverse Engineering AI Models

Kang Li
kangli.ctf@gmail.com

Collaborators： Deyue Zhang, Jiayu Qian, Yufei Chen

About Me

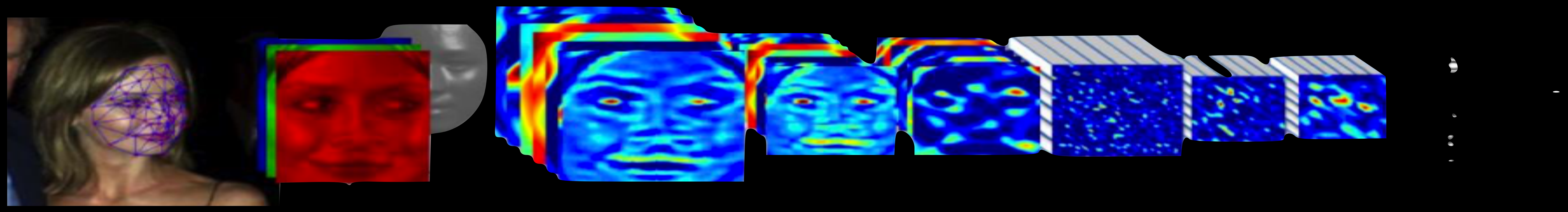
Professor of Computer Science at UGA

Founder of the SecDawgs, Disekt CTF Teams

Founding Mentor of xCTF and Blue-Lotus

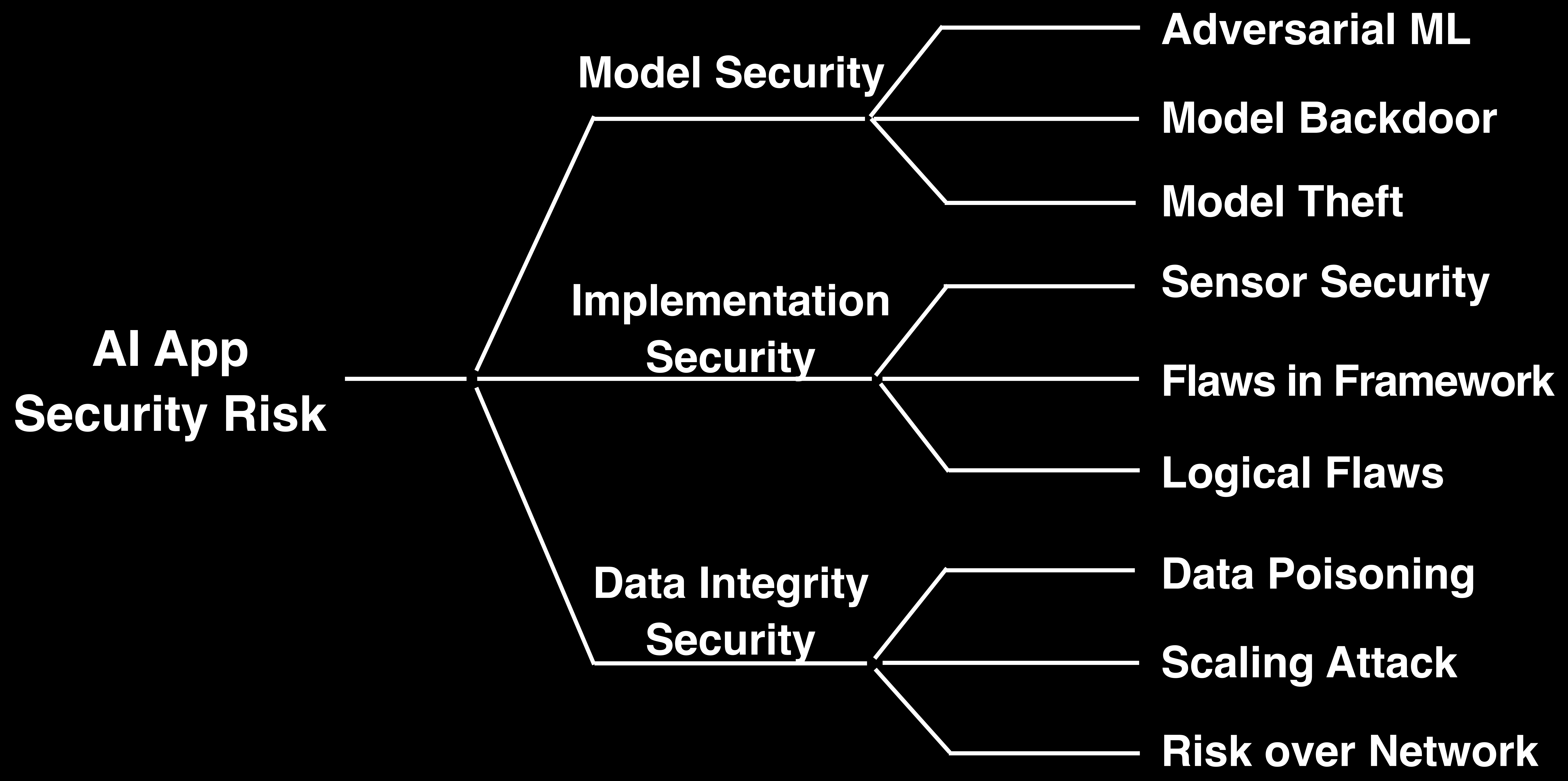
2016 DARPA Cyber Grand Challenge Finalist

Deep Learning AI Applications

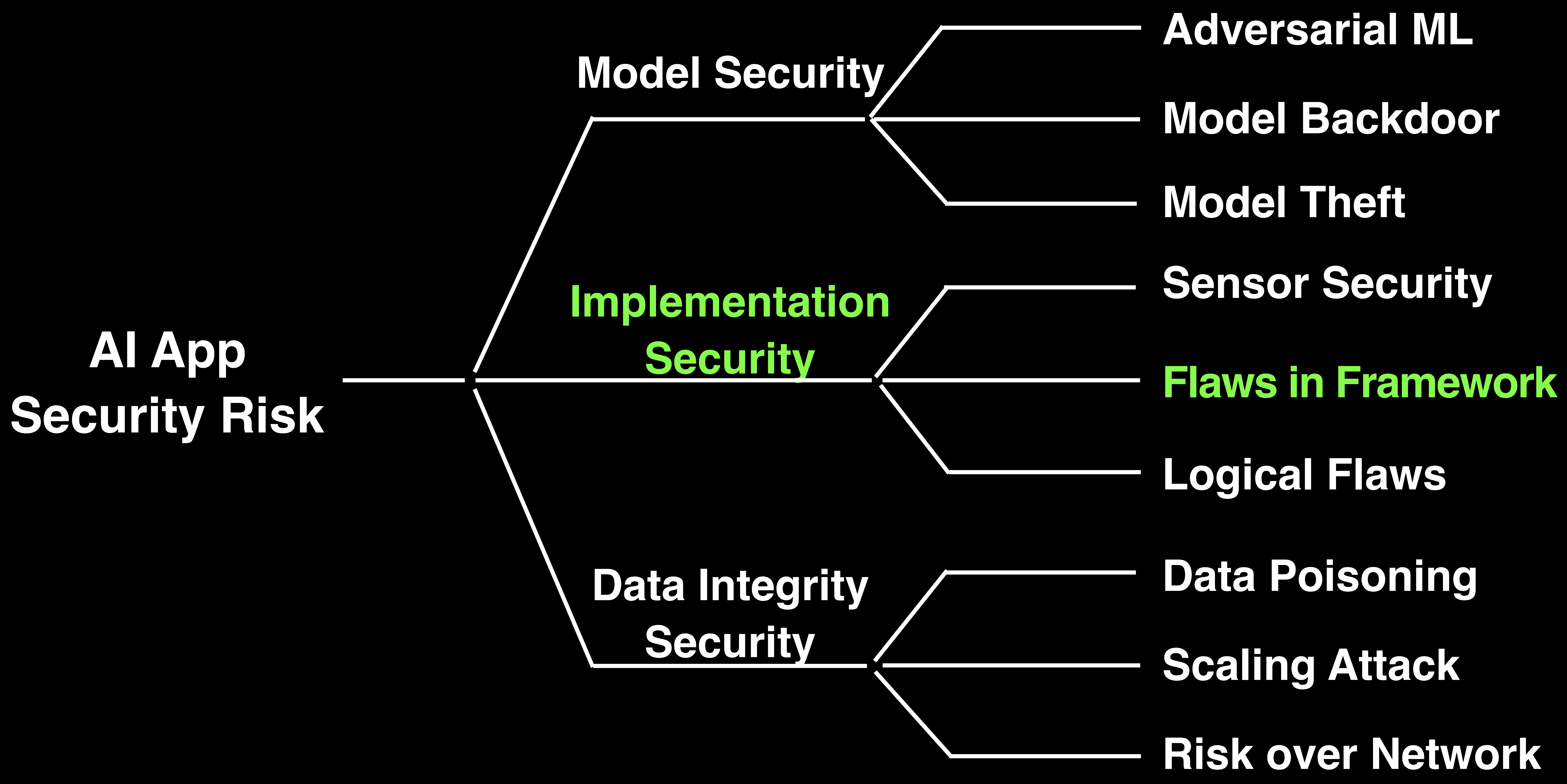


AI Application Security

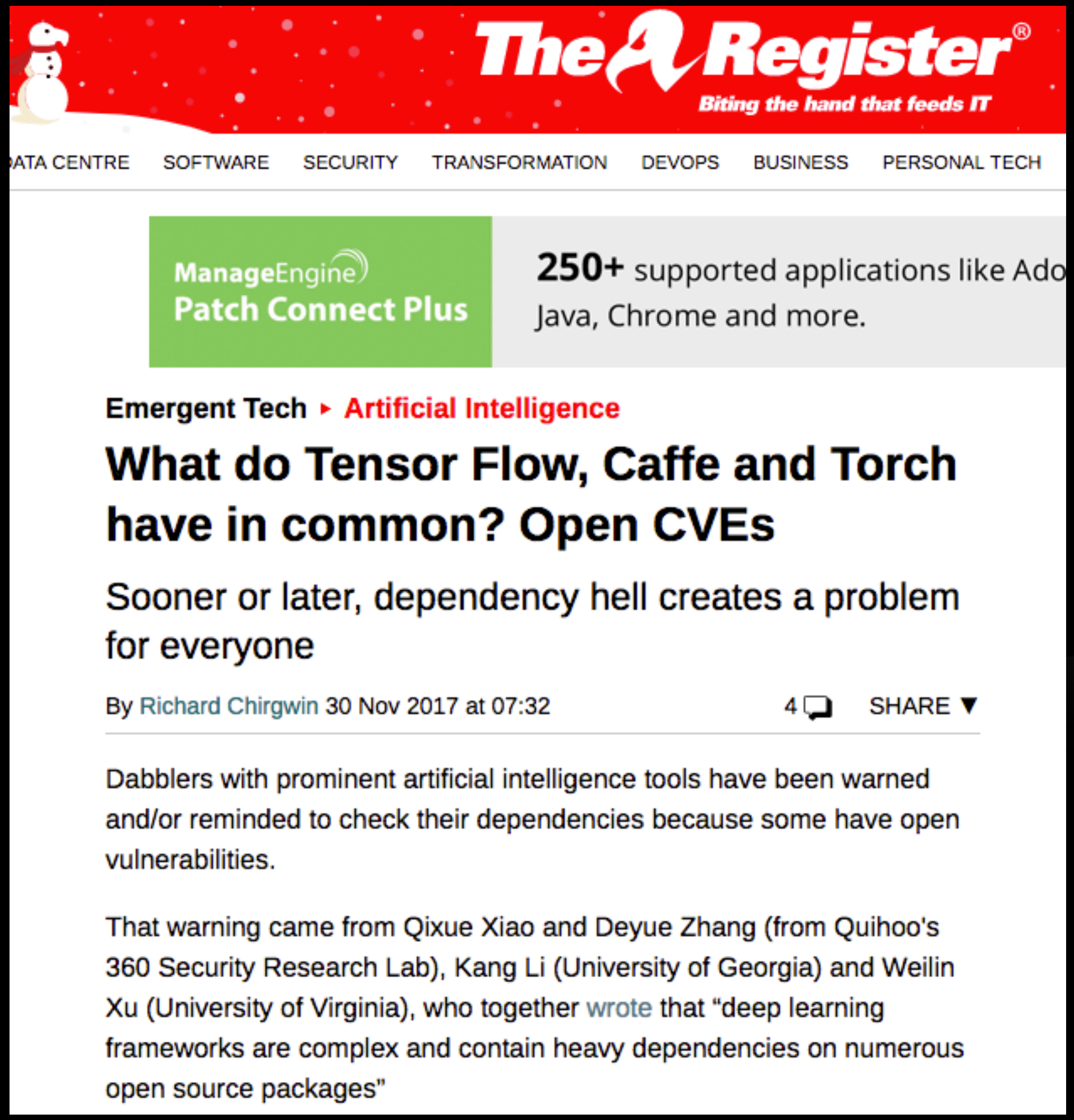
AI Application Threat Landscape (Kang's view)



My Prior Work



Security Risks in Deep Learning Implementations


The Register®
Biting the hand that feeds IT

ATA CENTRE SOFTWARE SECURITY TRANSFORMATION DEVOPS BUSINESS PERSONAL TECH

Emergent Tech ▸ Artificial Intelligence

What do Tensor Flow, Caffe and Torch have in common? Open CVEs

Sooner or later, dependency hell creates a problem for everyone

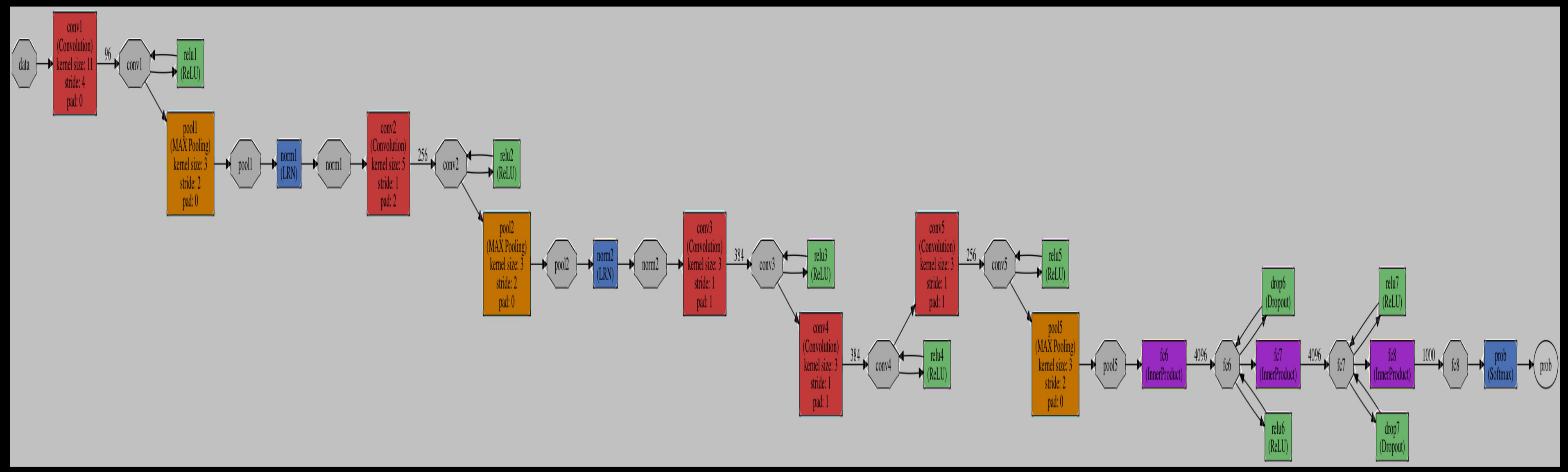
By Richard Chirgwin 30 Nov 2017 at 07:32 4 SHARE ▼

Dabblers with prominent artificial intelligence tools have been warned and/or reminded to check their dependencies because some have open vulnerabilities.

That warning came from Qixue Xiao and Deyue Zhang (from Quihoo's 360 Security Research Lab), Kang Li (University of Georgia) and Weilin Xu (University of Virginia), who together wrote that "deep learning frameworks are complex and contain heavy dependencies on numerous open source packages"

"Security Risks in Deep Learning Implementations", paper available at https://www.ieee-security.org/TC/SPW2018/DLS

Sample Deep Learning Application (CAFFE ImageNet)



```
./build/examples/cpp_classification/classification.bin \
  models/bvlc_reference_caffenet/deploy.prototxt \
  models/bvlc_reference_caffenet/bvlc_reference_caffenet.caffemodel \
  data/ilsvrc12/imagenet_mean.binaryproto \
  data/ilsvrc12/synset_words.txt \
  examples/images/cat.jpg
```

Output:

```
---------- Prediction for examples/images/cat.jpg ----------
0.3134 - "n02123045 tabby, tabby cat"
0.2380 - "n02123159 tiger cat"
0.1235 - "n02124075 Egyptian cat"
0.1003 - "n02119022 red fox, Vulpes vulpes"
0.0715 - "n02127052 lynx, catamount"
```

https://github.com/BVLC/caffe/tree/master/examples/cpp_classification

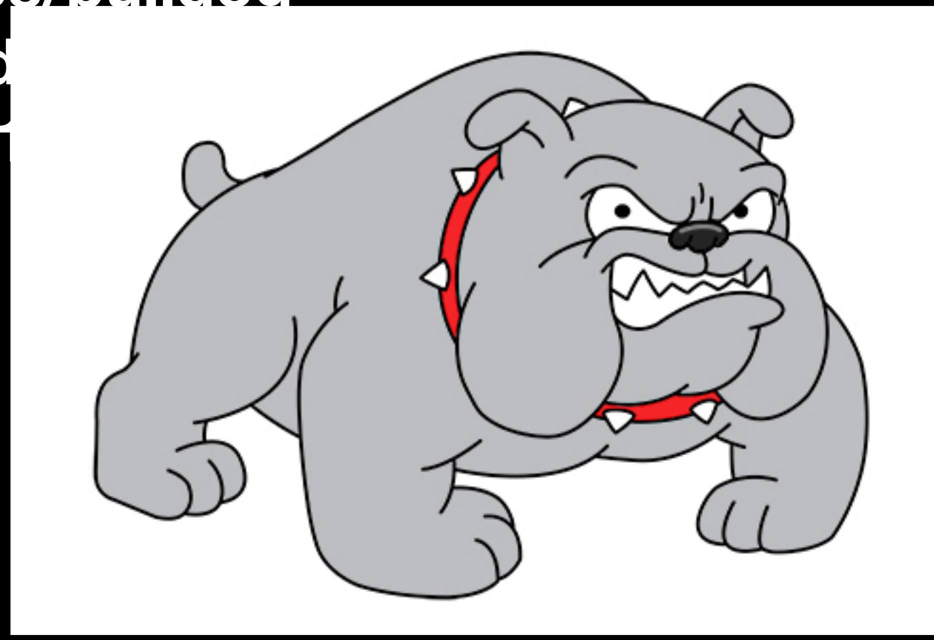
bullog [Original Image]
./classification.bin models/bvlc_reference_caffenet/deploy.prototxt
models/bvlc_reference_caffenet/bvlc_reference_caffenet.caffemodel
data/ilsvrc12/imagenet_mean.binaryproto
data/ilsvrc12/synset_words.txt
./poc_samples/bulldog
---------- Prediction for ./poc_samples/bulldog — — — — —
0.5111 - "n02108915 French bulldog"

Threat Example 2 -- Evasion attack
bulldog_sh [cause classification to misclassify to an arbitrary ca
[Here the classification produced a class "Flying Pig" made up b
./classification.bin models/bvlc_reference_caffenet/deploy.prototx
models/bvlc_reference_caffenet/bvlc_reference_caffenet.caffemod
data/ilsvrc12/imagenet_mean.binaryproto
data/ilsvrc12/synset_words.txt
./poc_samples/bulldog_fp
---------- Prediction for ./poc_samples/bulldog_fp ----------
0.98 - "n03770679 flyingpig”

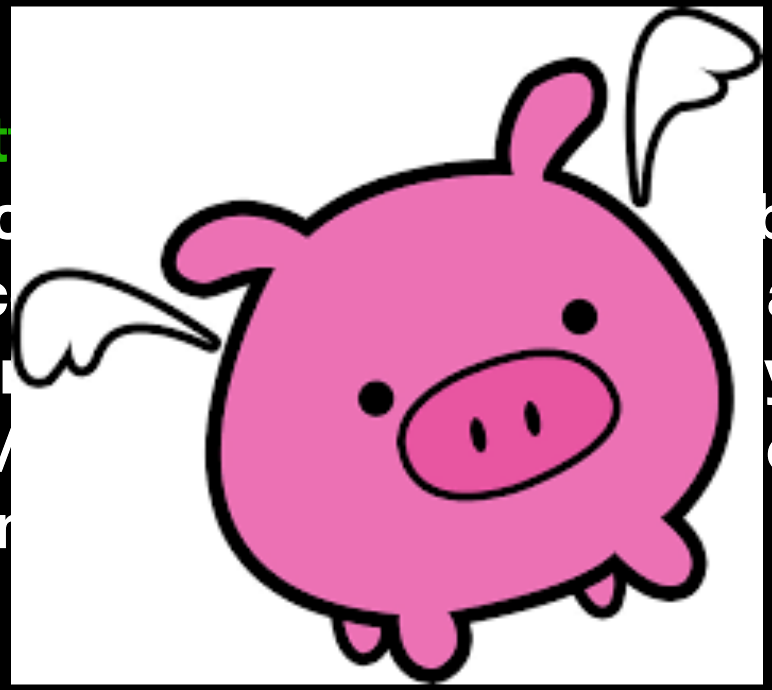
Threat Example 1 -- DoS attack
bulldog_crash [cause classification binary to crash]
./classification.bin models/bvlc_reference_caffenet/deploy.prototxt
models/bvlc_reference_caffenet/bvlc_reference_caffenet.caffemodel
data/ilsvrc12/imagenet_mean.binaryproto
data/ilsvrc12/synset_words.txt
./poc_samples/bulldog_crash
---------- Prediction for ./poc_samples/bulldog_crash ----------
Segmentation fault (core dumped)

Threat Example 3 -- Exploitation Attack
bulldog_sh [cause classification to generate a local shell]
./classification.bin models/bvlc_reference_caffenet/deploy.prototx
models/bvlc_reference_caffenet/bvlc_reference_caffenet.caffemod
data/ilsvrc12/imagenet_mean.binaryproto
data/ilsvrc12/synset_words.txt
./poc_samples/bulldog_sh
---------- Prediction for ./poc_samples/bulldog_sh ----------
$ uname -a
Linux ctf-box 4.4.0-31-generic #50~14.04.1-Ubuntu SMP
Wed Jul 13 01:07:32 UTC 2016 x86_64 x86_64 x86_64 GNU/Linux
$ exit

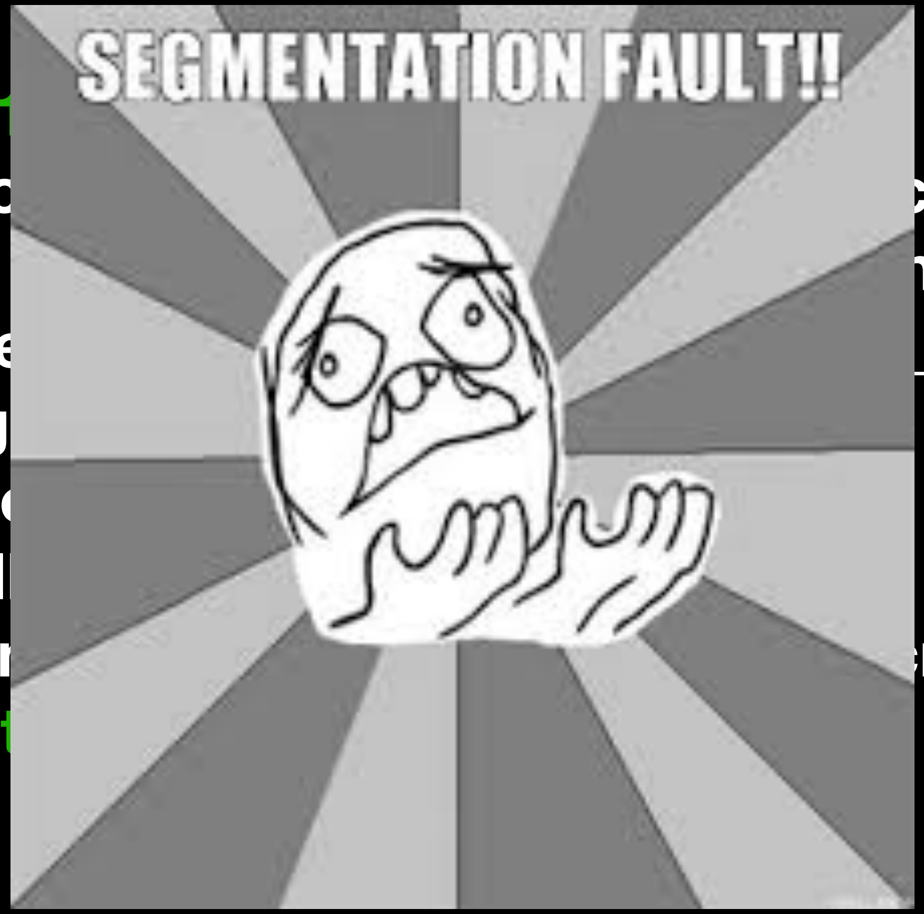
bullog [Original Image]
./classification.bin models/bvlc_reference_caffenet/deploy.prototxt
models/bvlc_reference_caffenet/bvlc_reference_caffenet.caffemodel
data/ilsvrc12/imagenet_mean.binaryproto
data/ilsvrc12/synset_words.txt
./poc_samples/bulldog
---------- Pred og — — — — —
0.5111 - "n021



Threat Example 2 -- Evasion at
bulldog_sh [cause classificati bitrary ca
[Here the classification produc ade up b
./classification.bin models/bvlc_ y.prototx
models/bvlc_reference_caffenet/ caffemod
data/ilsvrc12/imagenet_mean.bi
data/ilsvrc12/synset_words.txt
./poc_samples/bulldog_fp
---------- Prediction for ./poc_samples/bulldog_fp ----------
0.98 - "n03770679 flyingpig"

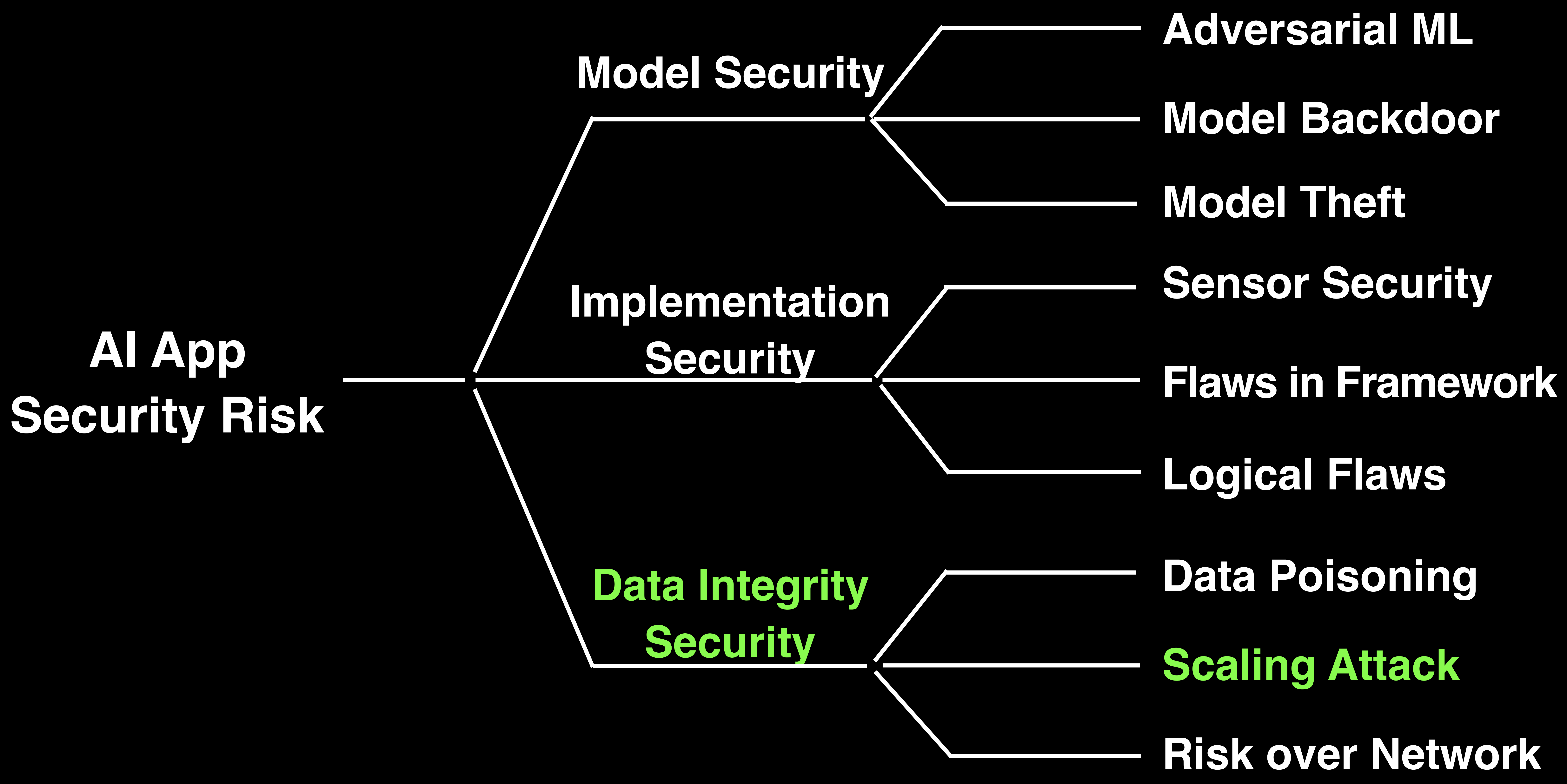


Threat Example 1
bulldog_crash [c crash]
./classification.bin net/deploy.prototxt
models/bvlc_refere caffenet.caffemodel
data/ilsvrc12/imag
data/ilsvrc12/syns
./poc_samples/bul
---------- Prediction rash ----------
Segmentation fault



Threat E k
bulldog_ nerate a local shell]
./classific ce_caffenet/deploy.prototx
models/bv ference_caffenet.caffemod
data/ilsvr to
data/ilsvr
./poc_sam
---------- ulldog_sh ----------
$ uname -
Linux ctf- .1-Ubuntu SMP
Wed Jul 1 6_64 x86_64 GNU/Linux
$ exit

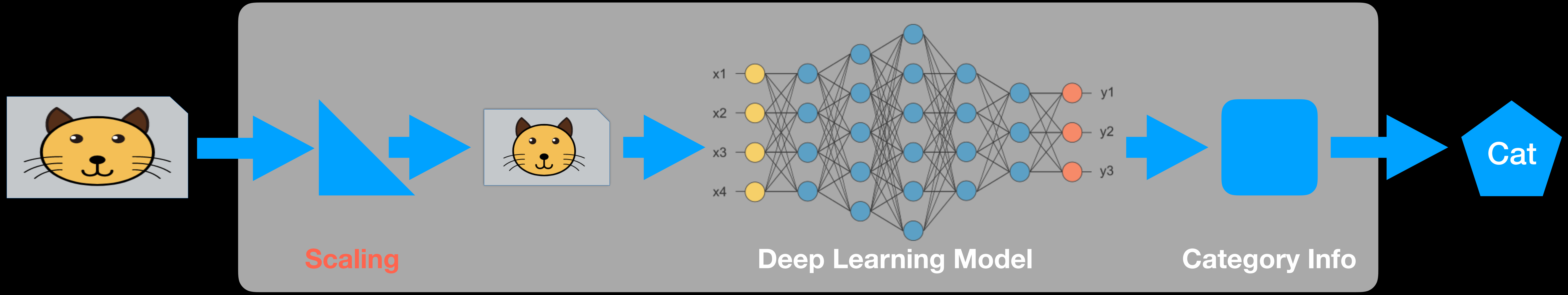
My Prior Work



Security Risks in Data Pre-Processing (Scaling)

Data Scaling Attacks in Deep Learning Applications
https://www.defcon.org/html/defcon-china/dc-cn-speakers.html#LiKang

Data Flow in Deep Learning Image Applications



Scaling Functions Provided by Frameworks

TensorFlow Example

```
def read_tensor_from_image_file(file_name, input_height=299, input_width=299,
                input_mean=0, input_std=255):
  input_name = "file_reader"
  output_name = "normalized"
  file_reader = tf.read_file(file_name, input_name)
  if file_name.endswith(".png"):
    image_reader = tf.image.decode_png(file_reader, channels = 3,
                                       name='png_reader')
  elif file_name.endswith(".gif"):
    image_reader = tf.squeeze(tf.image.decode_gif(file_reader,
                                                  name='gif_reader'))
  elif file_name.endswith(".bmp"):
    image_reader = tf.image.decode_bmp(file_reader, name='bmp_reader')
  else:
    image_reader = tf.image.decode_jpeg(file_reader, channels = 3,
                                        name='jpeg_reader')
  float_caster = tf.cast(image_reader, tf.float32)
  dims_expander = tf.expand_dims(float_caster, 0);
  resized = tf.image.resize_bilinear(dims_expander, [input_height, input_width])
  normalized = tf.divide(tf.subtract(resized, [input_mean]), [input_std])
  sess = tf.Session()
  result = sess.run(normalized)

  return result
```

tf.image.**resize_bilinear**(dims_expander, [input_height, input_width])

Attack Sample (Data Poisoning)

Evading Attack Example

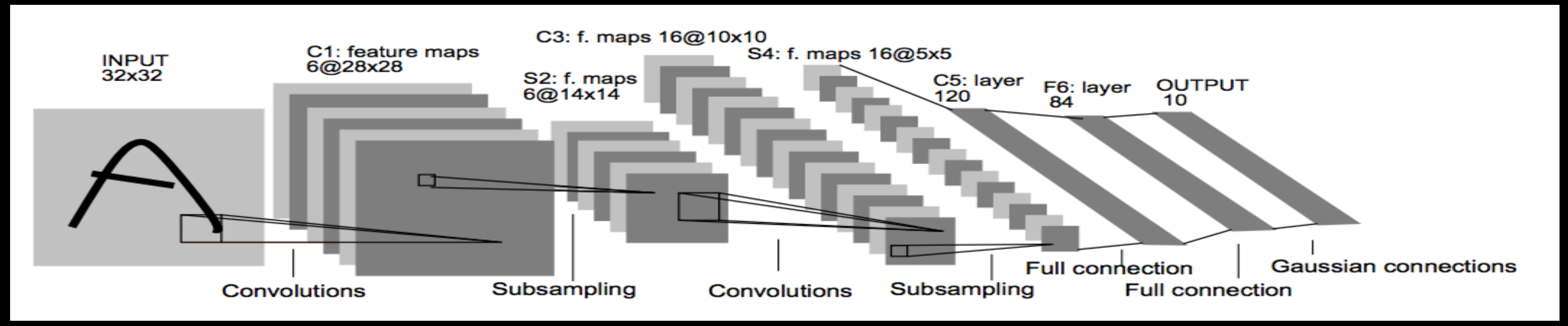
FEATURE NAME:	VALUE
Description	{ "tags": ["animal", "mammal", "wolf", "looking"], "captions": [{ "text": "a close up of a wolf", "confidence": 0.707954049 }] }
Tags	[{ "name": "animal", "confidence": 0.9989328 }, { "name": "mammal", "confidence": 0.9908992 }, { "name": "wolf", "confidence": 0.981169641 }]
Image format	"Jpeg"
Image dimensions	1024 x 1024
Clip art	0

- ✓ **Description: { "tags": ["animal", "mammal", "wolf", "looking"], "captions": [{ "text": "a close up of a wolf", "confidence": 0.707954049 }] }**
- ✓ **Tags: […, { "name": "wolf", "confidence": 0.981169641 }]**

AI Model Security

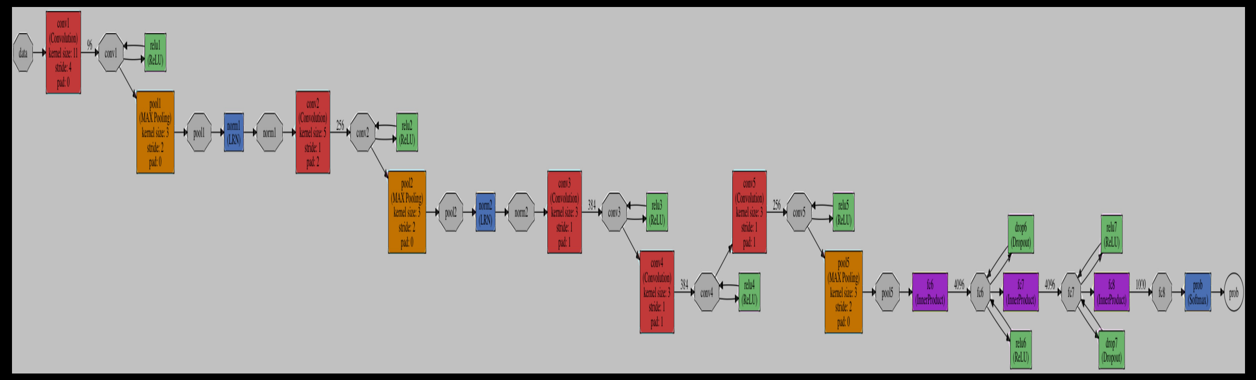
AI & Models

MNIST



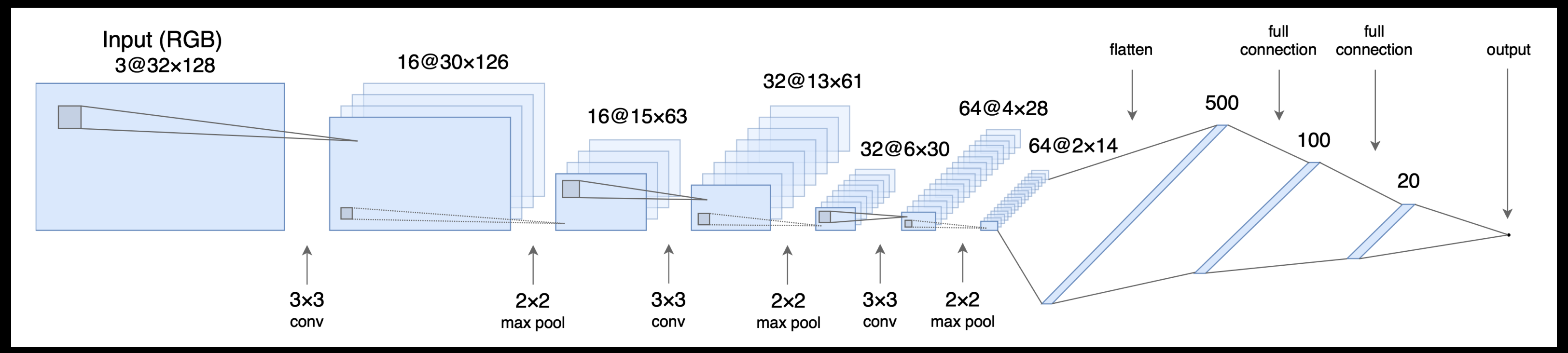
http://yann.lecun.com/exdb/publis/pdf/lecun-01a.pdf

ImageNet



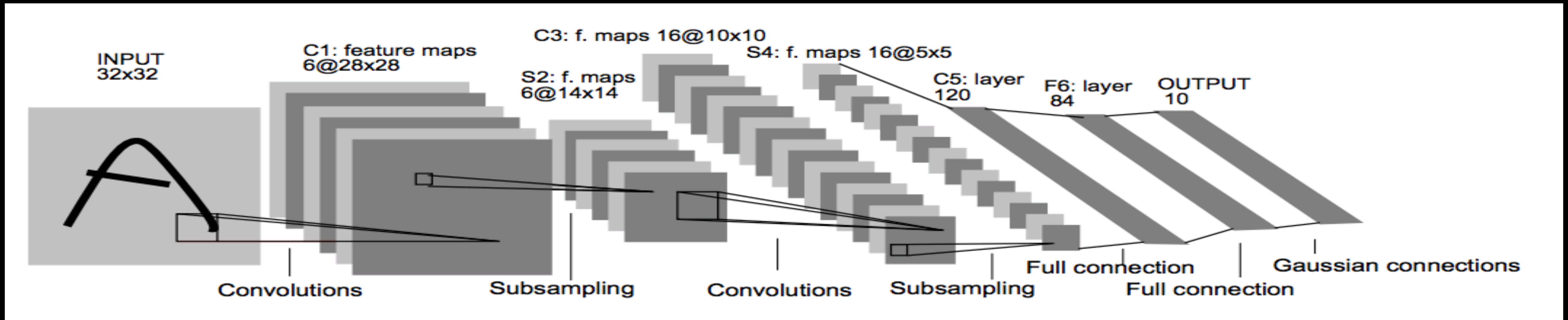
https://github.com/BVLC/caffe/tree/master/examples/cpp_classification

NVIDIA PX DAVE-2



https://images.nvidia.com/content/tegra/automotive/images/2016/solutions/pdf/end-to-end-dl-using-px.pdf

Sample Neural Network (LeNet-5) Architecture

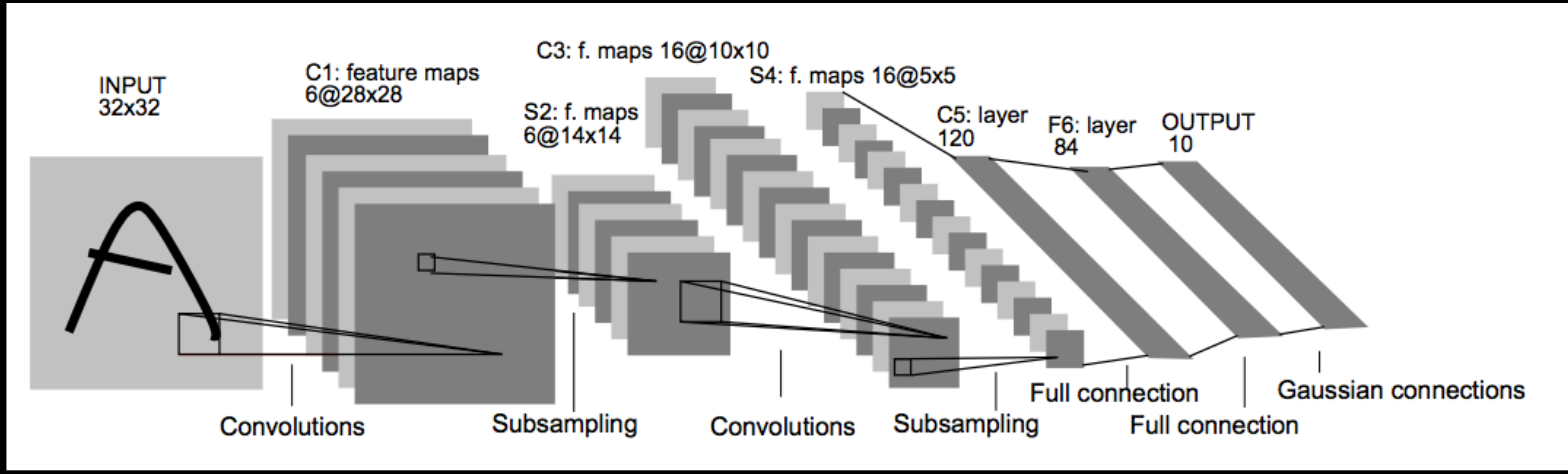


http://yann.lecun.com/exdb/publis/pdf/lecun-01a.pdf

Sample Model Files

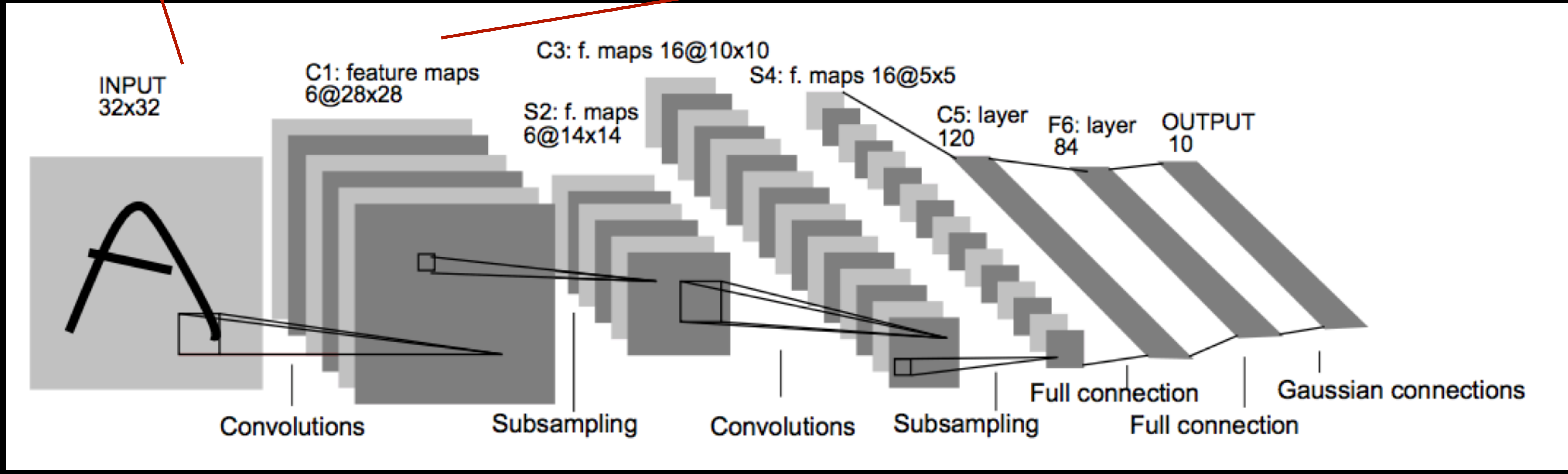
```
name: "LeNet"
layer {
  name: "data"
  type: "Input"
  top: "data"
  input_param
{ shape: { dim: 1
dim: 1 dim: 28 dim:
28 } }
}
layer {
  name: "conv1"
  type: "Convolution"
  bottom: "data"
  top: "conv1"
  param {
    lr_mult: 1
  }
  param {
    lr_mult: 2
  }
  convolution_param
{
    num_output: 20
    kernel_size: 5
    stride: 1
    weight_filler {
      type: "xavier"
    }
    bias_filler {
      type: "constant"
    }
  }
}
layer {
  name: "pool1"
  type: "Pooling"
  bottom: "conv1"
  top: "pool1"
  pooling_param {
    pool: MAX
    kernel_size: 2
    stride: 2
  }
}
layer {
  name: "conv2"
  type: "Convolution"
  bottom: "pool1"
  top: "conv2"
  param {
    lr_mult: 1
  }
  param {
    lr_mult: 2
  }
  convolution_param
{
    num_output: 50
    kernel_size: 5
    stride: 1
    weight_filler {
      type: "xavier"
    }
    bias_filler {
      type: "constant"
    }
  }
}
layer {
  name: "pool2"
  type: “Pooling"
  bottom: "conv2"
  top: "pool2"
  pooling_param {
    pool: MAX
    kernel_size: 2
    stride: 2
  }
}
layer {
  name: "ip1"
  type:
"InnerProduct"
  bottom: "pool2"
  top: "ip1"
  param {
    lr_mult: 1
  }
  param {
    lr_mult: 2
  }

inner_product_para
m {
    num_output: 500
    weight_filler {
      type: "xavier"
    }
    bias_filler {
      type: "constant"
    }
  }
}
layer {
  name: "relu1"
  type: "ReLU"
  bottom: "ip1"
  top: "ip1"
}
layer {
  name: "ip2"
  type:
"InnerProduct"
  bottom: "ip1"
  top: "ip2"
  param {
    lr_mult: 1
  }
  param {

    lr_mult: 2
  }

inner_product_para
m {
    num_output: 10
    weight_filler {
      type: "xavier"
    }
    bias_filler {
      type: "constant"
    }
  }
}
layer {
  name: "prob"
  type: "Softmax"
  bottom: "ip2"
  top: "prob"
}
```

Model Layers (lenet_deploy.prototxt)



Sample Model Files

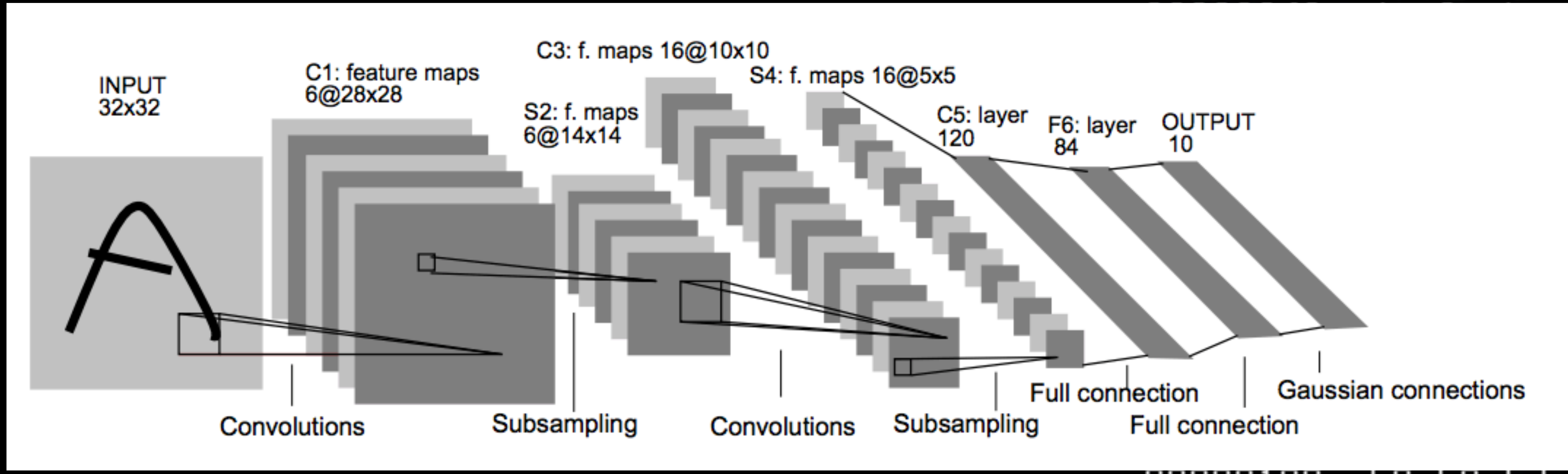
```
name: "LeNet"
layer {
  name: "data"
  type: "Input"
  top: "data"
  input_param { shape: { dim: 1 dim: 1 dim: 28 dim: 28 } }
}
```




```
layer {
  name: "conv1"
  type: "Convolution"
  bottom: "data"
  top: "conv1"
  param {
    lr_mult: 1
  }
  param {
    lr_mult: 2
  }
  convolution_param {
    num_output: 20
    kernel_size: 5
    stride: 1
    weight_filler {
      type: "xavier"
    }
    bias_filler {
      type: "constant"
    }
  }
}
```

Sample Model Files

Model Parameters
(lenet.caffemodel)




```
00000000  0a 05 4c 65 4e 65 74 a2  06 50 0a 05 6d 6e 69 73  |..LeNet..P..mnis|
00000010  74 12 04 44 61 74 61 22  04 64 61 74 61 22 05 6c  |t..Data".data".l|
00000020  61 62 65 6c 42 02 08 00  50 00 a2 06 05 0d 00 00  |abelB...P.......|
00000030  80 3b da 06 25 0a 1f 65  78 61 6d 70 6c 65 73 2f  |.;..%..examples/|
00000040  6d 6e 69 73 74 2f 6d 6e  69 73 74 5f 74 72 61 69  |mnist/mnist_trai|
00000050  6e 5f 6c 6d 64 62 20 40  40 01 a2 06 87 11 0a 05  |n_lmdb @@.......|
00000060  63 6f 6e 76 31 12 0b 43  6f 6e 76 6f 6c 75 74 69  |conv1..Convoluti|
00000070  6f 6e 1a 04 64 61 74 61  22 05 63 6f 6e 76 31 32  |on..data".conv12|
00000080  05 1d 00 00 80 3f 32 05  1d 00 00 00 40 3a db 0f  |.....?2.....@:..|
00000090  2a d0 0f 20 a4 82 3d 57  98 d5 3e 35 24 76 3e 15  |*.. ..=W..>5$v>.|
000000a0  01 ed 3e ed ce 90 3d 33  34 b0 3e 2c b8 85 3e 4e  |..>...=34.>,..>N|
000000b0  35 11 3e ae d1 b0 3d 79  81 22 3e d8 48 b3 be 26  |5.>...=y.">.H..&|
000000c0  fd 19 be 3f 75 71 be f5  a9 69 be 72 12 cf be ce  |...?uq...i.r....|
000000d0  02 a5 be 4d fd 6f be 49  26 8f be 68 c4 91 be d1  |...M.o.I&..h....|
000000e0  d1 bf bd ab 1f cc be cb  9b 8f be 89 f7 c4 be 12  |................|
                           7a 2f ef bc 3e  e9 74 3e a8 ae 07 3e 8a  |J>.z/..>.t>...>.|
                           8a 20 53 bd 1b  75 bc be 3a c3 2d 3e cd  |.;>. S..u..:.->.|
                           f0 f4 3e be ab  21 ed be 04 45 29 bd 13  |..<..>..!...E)..|
                           8f 01 80 bd e0  3f 79 be 86 4d 62 bc a4  |.}=.....?y..Mb..|
                           d5 00 94 3d e1  d8 bf bc 5a d7 cd 3e cc  |j.>...=....Z..>.|
                           0c 77 9e 3e 1e  ed 8f 3e f4 ea a0 3e f6  |..>.w.>...>...>.|
                           04 64 7c 3e b5  d8 ff bd 66 56 13 be cc  |..>.d|>....fV...|
                           8e 2d 02 3c 80  46 83 bd 7a 9e c9 bc f8  |.m>.-.<.F..z....|
                           85 48 19 3e 85  df c4 bd 02 f4 73 3e fe  |C...H.>......s>.|
00000180  b0 b8 bd f5 6f 9c be c0  8d e9 3a 6a 5f a2 3e 07  |....o.....:j_.>.|
00000190  69 97 3e c4 3a a1 3e 2c  3a 7e be c5 36 52 bd f5  |i.>.:.>,:~..6R..|
```

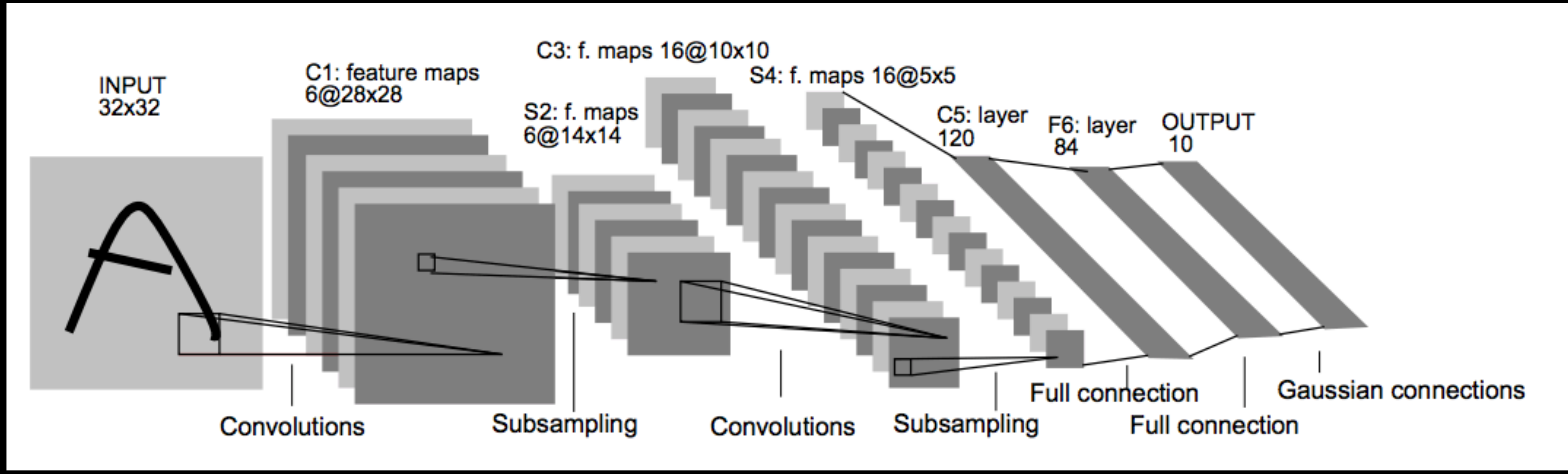
Sample Model Files

Model Parameters
(lenet.caffemodel)

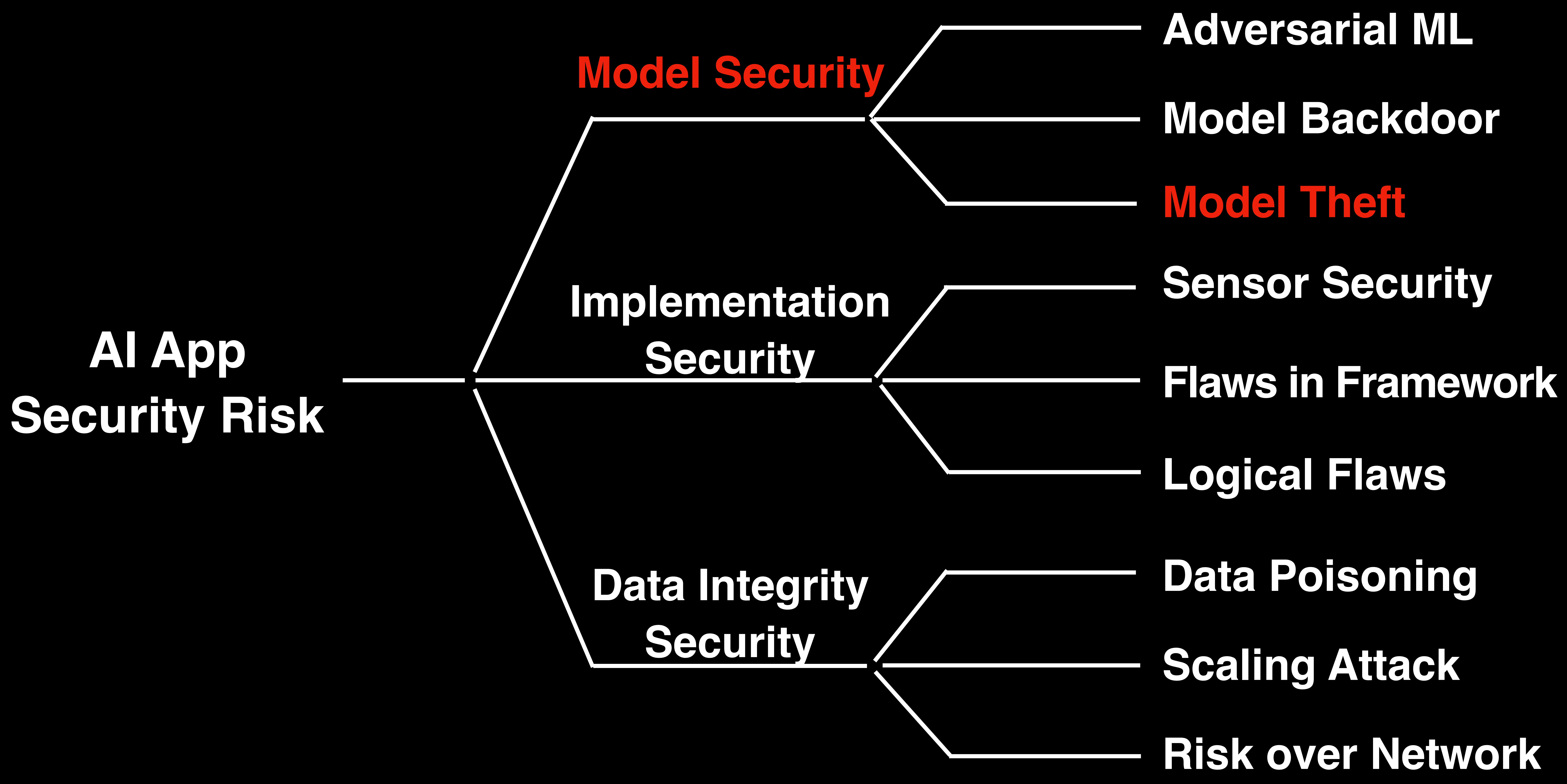
Parameters translated to
plain text format

```
layer {
  name: "conv1"
  type: "Convolution"
  bottom: "data"
  top: "conv1"
  param {
    lr_mult: 1
  }
  param {
    lr_mult: 2
  }
  blobs {
    data: 0.063789606
    data: 0.41717789
    data: 0.24037249
     ... ...
```

```
 data: 0.0044610952
    shape {
      dim: 10
      dim: 500
    }
  }
  blobs {
    data: -0.0088488162
    data: -0.0015859469
    data: -0.015499314
  ... ...
```



AI Application Security Threats



Threat related to AI Models

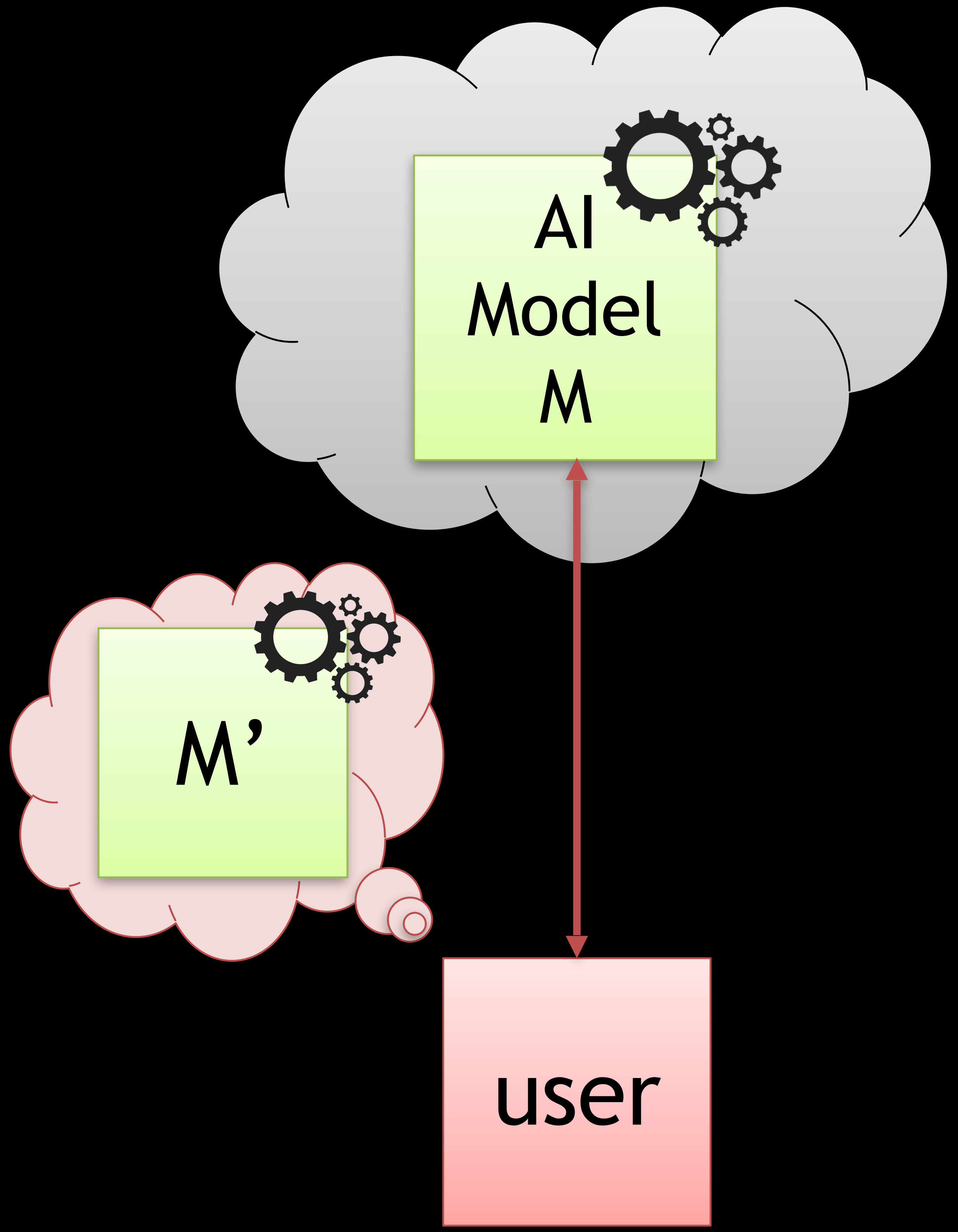
Theory

- model inference attack
- model inverse attack
- membership inference attack

Reality

- Reverse engineering attack
- software vulnerability

Attacks Related to Deep Learning Models

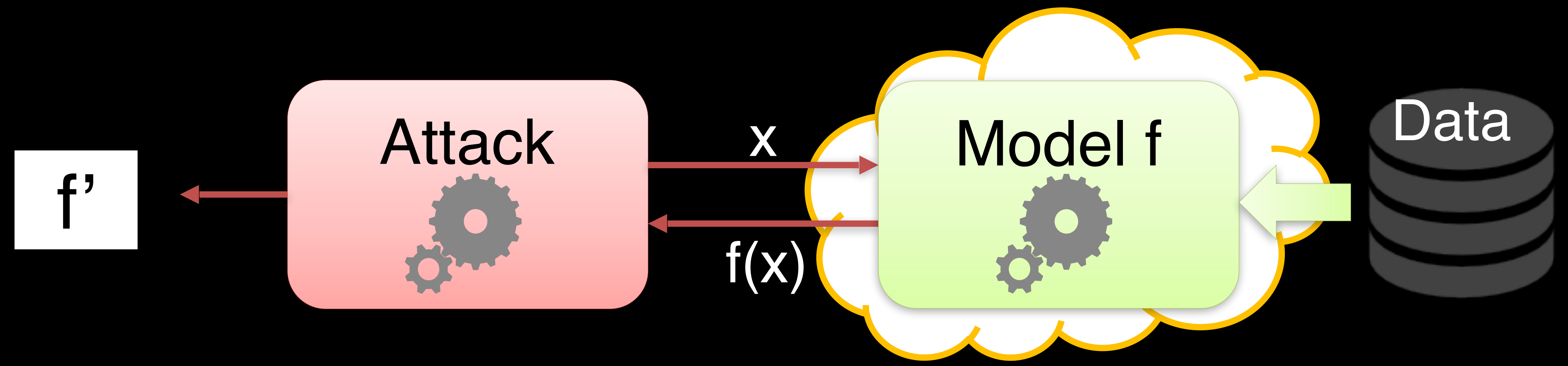


- Valuable Assets of AI Applications
  - Models and Parameters (and Training data)
- Academics have paid substantial attentions at the danger of models and parameters leaking.

F. Tramèr, F. Zhang, A. Juels, M. K. Reiter, and T. Ristenpart, “Stealing Machine Learning Models via Prediction APIs,” ***USENIX Security ’16***.

https://github.com/ftramer/Steal-ML

Model Inference Attack

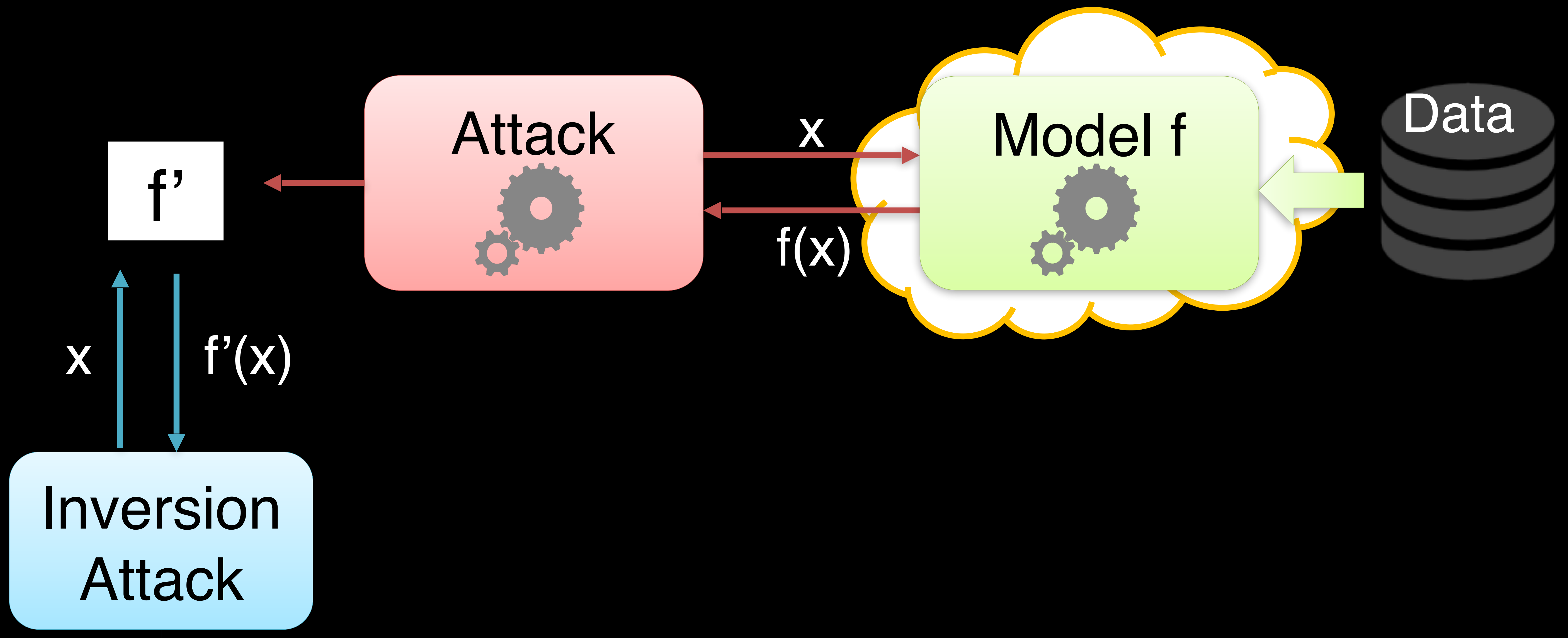


- Attack on cloud-based AI models
  - Selectively construct queries to Infer algorithms and parameters
- Targeted Algorithms
  - *Logistic Regressions, SVMs, Decision Trees*

F. Tramèr, F. Zhang, A. Juels, M. K. Reiter, and T. Ristenpart, "Stealing Machine Learning Models via Prediction APIs," ***USENIX Security '16***.

https://github.com/ftramer/Steal-ML

Model Inversion Attack



- Infer information related to training data through specially crafted queries

F. Tramèr, F. Zhang, A. Juels, M. K. Reiter, and T. Ristenpart, "Stealing Machine Learning Models via Prediction APIs," ***USENIX Security '16***.

https://github.com/ftramer/Steal-ML

Example: Model Inversion Attack

Training Images

Reconstructed Images

- Goal: to infer information in training data
- Experiments for Model Inversion Attack
  - Photo Images from 40 people
  - Simple Neural Network（one hidden layer）

M. Fredrikson, S. Jha, and T. Ristenpart, “Model Inversion Attacks that Exploit Confidence Information and Basic Countermeasures,” ***ACM CCS 2015***.

Other Academic Works

- Inferring deep learning hyper-parameters

Binghui Wang and Neil Zhenqiang Gong. “Stealing Hyperparameters in Machine Learning”. In *IEEE Symposium on Security and Privacy*, 2018.

- Inferring membership info in machine learning training data

R. Shokri, M. Stronati, C. Song, and V. Shmatikov, “Membership inference attacks against machine learning models,” in IEEE S & P, 2017.

C. Song, T. Ristenpart, and V. Shmatikov, “Machine learning models that remember too much,” in CCS, 2017

Threat related to AI Models

Theory

- model inference attack
- model inverse attack
- membership inference attack

Reality

- reverse engineering attack
- software vulnerability

Some APPs that use AI Models

Caffe

Caffe2

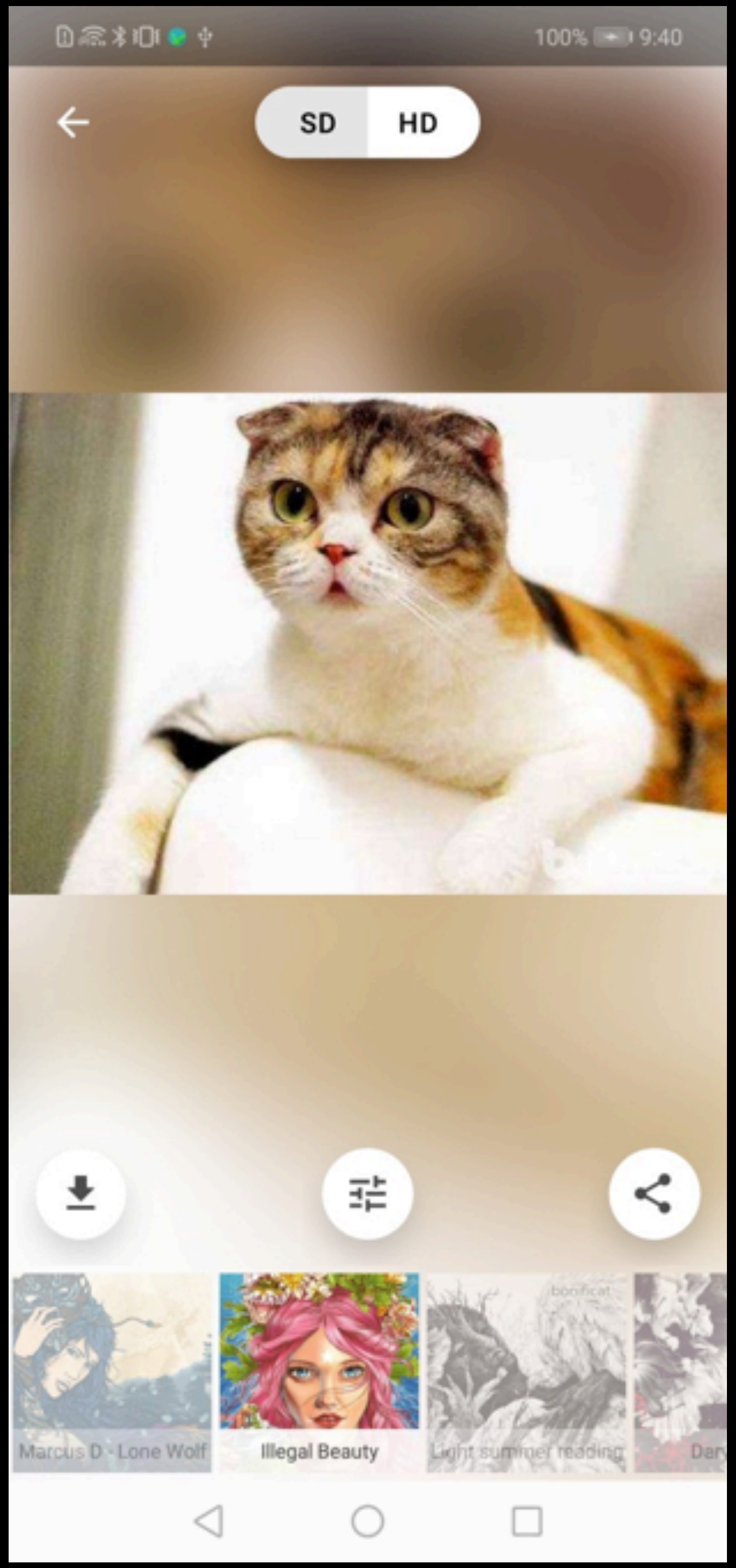
TF Lite

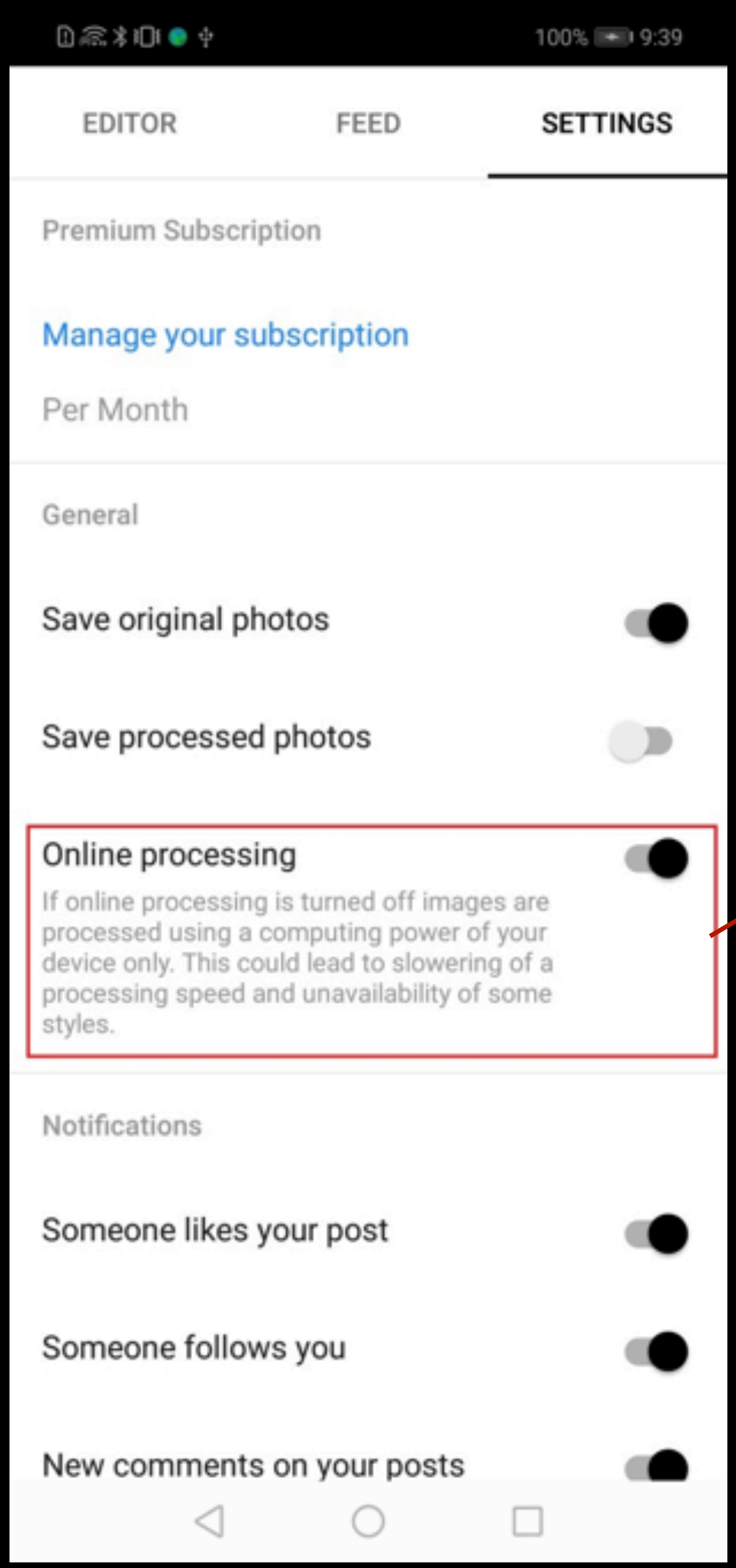
SenseTime

TensorFlow Mobile

MegVii

Mixed Platform





If online processing is turned off images are processed using a computing power of your device only. This could lead to lowering of a processing speed and unavailability of some styles.

😮 Local AI Models ?!

AI Model Files Found

- Locally stored models in APKs
  - Plain framework model files (*.caffemodel, *.pb)
  - Plain but "obfuscated" filenames (e.g *.mp3)
  - Encrypted model files
    - often using AES, with hard-coded but hidden keys
  - Model files for specific AI hardware (*.dlc , *.cambricon, *.pie)
- On-demand, model files downloaded from network

Local Model Files

- Built-in Model Files

```
APK
├─── AndroidManifest.xml
├─── assets
├───     ├─── 13.cambricon
├───     ├─── 15.cambricon
├───     ├─── 23.cambricon
├───     ├─── 24.cambricon
├───     ├─── 2.cambricon
```

```
APK    com.——.camera
├─── AndroidManifest.xml
├─── assets
├───     ├─── mdoel6.zip
```

- Download at runtime

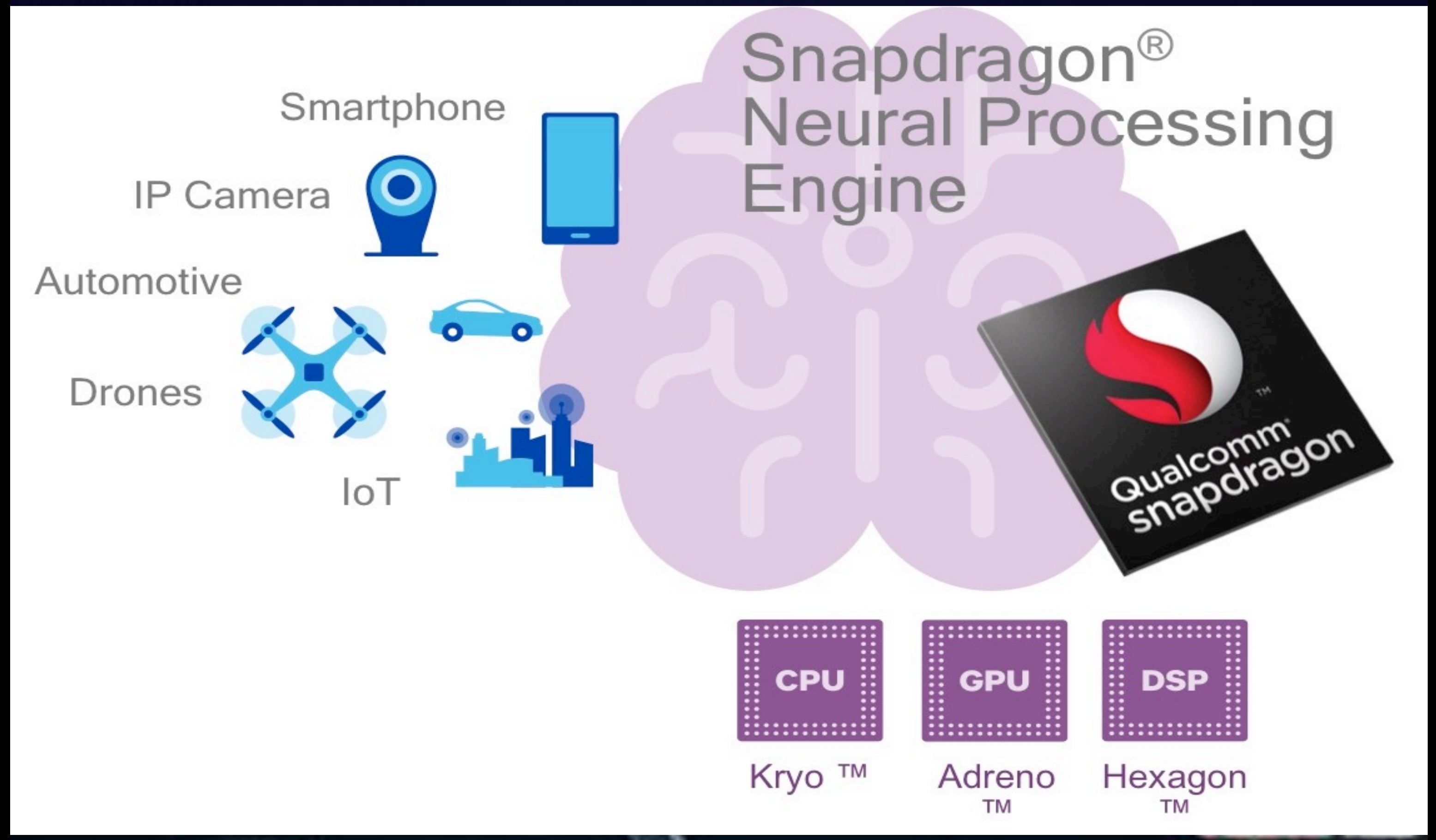
/sdcard/toffee/facemodels

/$APK_HOME/cache/temp_pies/illegal_beauty2.pie

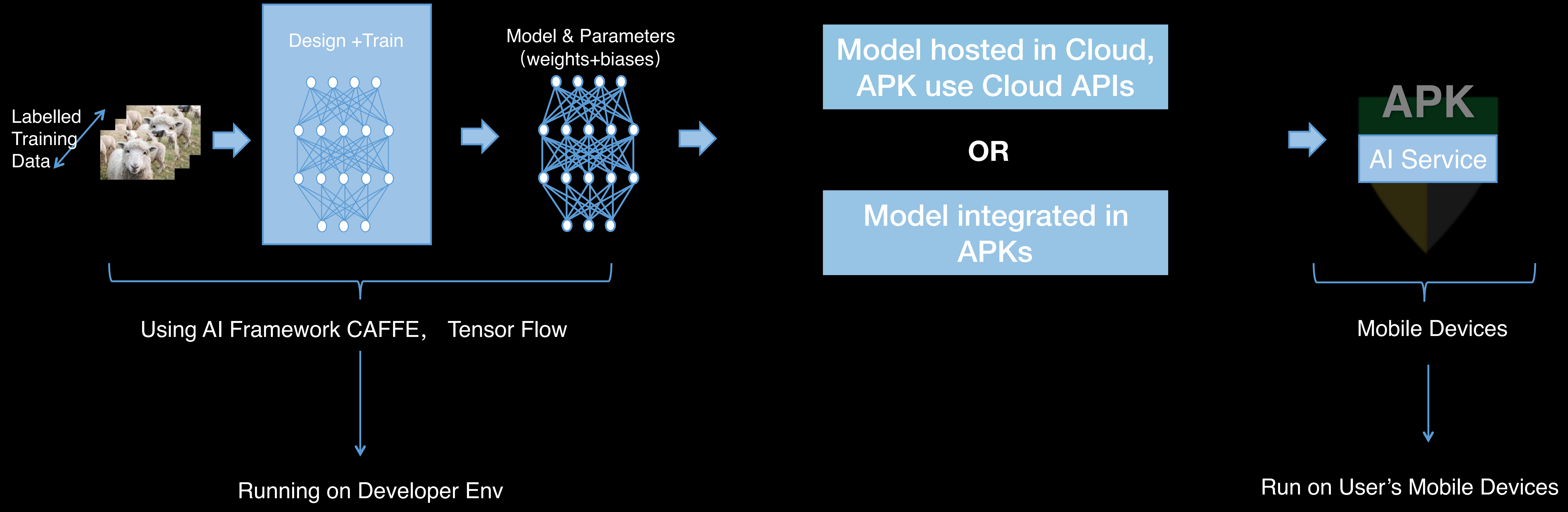

```
0000h: 06 00 00 00 73 74 64 63 6E 6E 03 00 00 00 80 00  ....stdcnn....€.
0010h: 00 00 80 00 00 00 2A 00 00 00 01 00 00 00 00 00  ..€...*.........
0020h: 00 00 01 00 00 00 01 00 00 00 01 00 00 00 02 00  ................
0030h: 00 00 01 00 00 00 03 00 00 00 01 00 00 00 04 00  ................
0040h: 00 00 01 00 00 00 05 00 00 00 01 00 00 00 06 00  ................
0050h: 00 00 01 00 00 00 07 00 00 00 01 00 00 00 08 00  ................
0060h: 00 00 01 00 00 00 09 00 00 00 01 00 00 00 0A 00  ................
0070h: 00 00 01 00 00 00 0B 00 00 00 01 00 00 00 0A 00  ................
0080h: 00 00 01 00 00 00 0D 00 00 00 01 00 00 00 0C 00  ................
0090h: 00 00 01 00 00 00 0F 00 00 00 01 00 00 00 0C 00  ................
00A0h: 00 00 01 00 00 00 11 00 00 00 01 00 00 00 0C 00  ................
00B0h: 00 00 01 00 00 00 13 00 00 00 01 00 00 00 0C 00  ................
00C0h: 00 00 01 00 00 00 15 00 00 00 04 00 00 00 10 00  ................
00D0h: 00 00 12 00 00 00 14 00 00 00 16 00 00 00 01 00  ................
00E0h: 00 00 0E 00 00 00 01 00 00 00 18 00 00 00 01 00  ................
00F0h: 00 00 0E 00 00 00 01 00 00 00 1A 00 00 00 01 00  ................
0100h: 00 00 0E 00 00 00 01 00 00 00 1C 00 00 00 03 00  ................
0110h: 00 00 19 00 00 00 1B 00 00 00 1D 00 00 00 01 00  ................
0120h: 00 00 17 00 00 00 01 00 00 00 1F 00 00 00 01 00  ................
0130h: 00 00 20 00 00 00 01 00 00 00 1E 00 00 00 01 00  .. .............
0140h: 00 00 22 00 00 00 01 00 00 00 23 00 00 00 01 00  ..".......#.....
0150h: 00 00 24 00 00 00 01 00 00 00 25 00 00 00 01 00  ..$.......%.....
0160h: 00 00 24 00 00 00 01 00 00 00 27 00 00 00 01 00  ..$.......'.....
0170h: 00 00 21 00 00 00 01 00 00 00 29 00 00 00 03 00  ..!.......).....
0180h: 00 00 26 00 00 00 28 00 00 00 2A 00 00 00 2C 00  ..&...(...*...,.
0190h: 00 00 74 79 70 65 3A 63 6F 6E 76 2C 6B 73 69 7A  ..type:conv,ksiz
01A0h: 65 3A 33 2C 73 74 72 69 64 65 3A 32 2C 70 61 64  e:3,stride:2,pad
01B0h: 3A 30 2C 6D 61 70 5F 6E 75 6D 3A 32 34 2C A0 02  :0,map_num:24, .
01C0h: 00 00 78 FD C4 BE D2 2F A0 BD 27 46 DB 3E 2A 6B  ..xýÄ¾Ò/ ½'FÛ>*k
01D0h: E6 BE 25 E2 A0 BD 29 9B D8 3E 7C 47 96 BE 37 1E  æ¾%â ½)›Ø>|G–¾7.
01E0h: E2 3D A4 C8 A1 3E F0 6D E9 BE 2E 77 80 BC 1F D6  â=¤È¡>ðmé¾.w€¼.Ö
```

Why Local AI Models ?

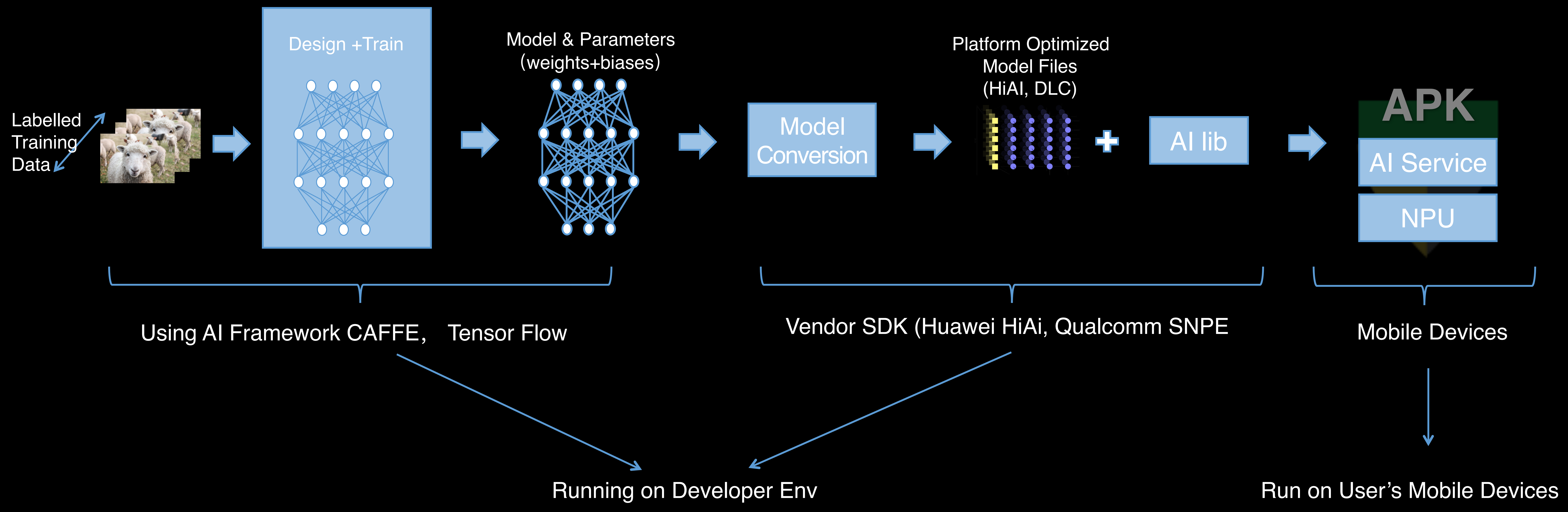
- AI processing is a new trend in the mobile world
  - Qualcomm NPE, Huawei HiAI, Samsung Exynos AI
- Major driven factors
  - Delay, bandwidth, privacy



Dev Process of AI Mobile App



Dev Process of AI Mobile App



Threat related to AI Models

Theory

- model inference attack
- model inverse attack
- membership inference attack

Reality

- reverse engineering attack
- software vulnerability

Reverse Engineering Example #1 (APP-A)

```
v5 = (void *)operator new[](0x10u);
v6 = "498[redacted]720";
v7 = v5;
while ( 1 )
```

```
}
ptr = malloc(_R5);
fread(ptr, 1u, _R5, v21);
aes_decrypt(128, (int)v53, v51, SHIDWORD(v51), (int)&ptr, (int)v53);
v26 = operator new[](8u);
*(_BYTE *)v26 = *(_BYTE *)ptr;
```

Model encrypted with AES,
hardcoded key in APK

Reverse Engineering Example #1 （APP-A)

```
v5 = (void *)operator new[](0x10u);
v6 = "498[redacted]720";
v7 = v5;
while ( 1 )
```

```
}
ptr = malloc(_R5);
fread(ptr, 1u, _R5, v21);
aes_decrypt(128, (int)v53, v51, SHIDWORD(v51), (int)&ptr, (int)v53);
v26 = operator new[](8u);
*(_BYTE *)v26 = *(_BYTE *)ptr;
```

Model encrypted with AES,
hardcoded key in APK

Script to extract model

```
#!/usr/bin/env python
from Crypto.Cipher import AES
import sys

key = "498[redacted]".decode('hex')
crypto = AES.new(key,AES.MODE_ECB,key)
data = ''
with open(sys.argv[1],'rb') as f:
        data = f.read()
with open(sys.argv[2],'wb') as f:
        data = data[16:]
        length = len(data)
        cycle = (length + 15) / 16
        for i in range(cycle):
                cipher = data[i*16 : (i+1)*16].ljust(16,'\x00')
                text = crypto.decrypt(cipher)
                f.write(text)
```

sample outcome

decrypted model file
(caffemodel)

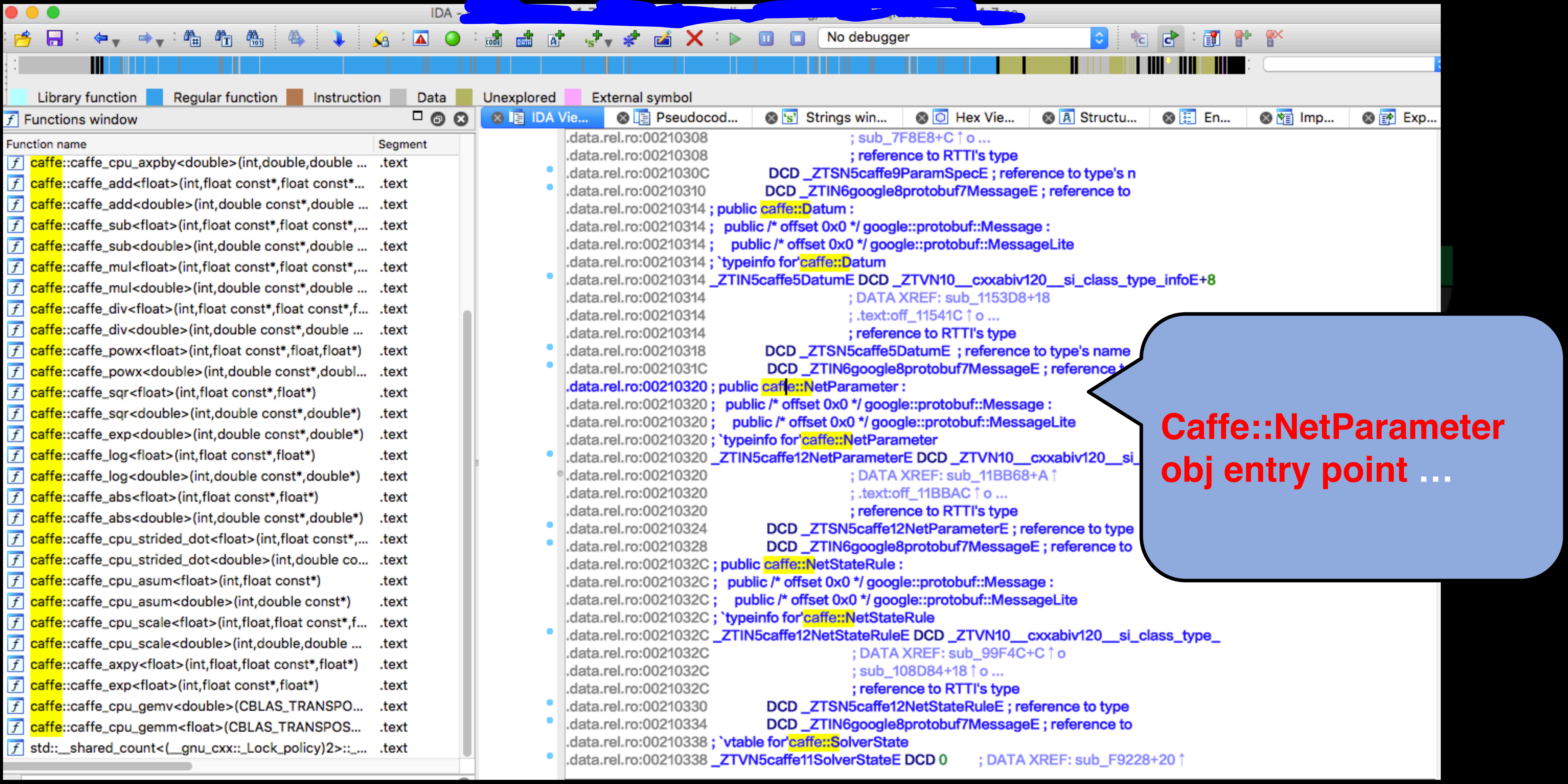
```
00000000  35 30 30 33 00 00 00 00  6e 61 6d 65 3a 20 22 54  |5003....name: "T|
00000010  43 44 43 4e 22 0a 6c 61  79 65 72 20 7b 0a 20 20  |CDCN".layer {.  |
00000020  6e 61 6d 65 3a 20 22 64  61 74 61 22 0a 20 20 74  |name: "data".  t|
00000030  79 70 65 3a 20 22 49 6e  70 75 74 22 0a 20 20 74  |ype: "Input".  t|
00000040  6f 70 3a 20 22 64 61 74  61 22 0a 20 20 69 6e 70  |op: "data".  inp|
00000050  75 74 5f 70 61 72 61 6d  20 7b 0a 20 20 20 20 73  |ut_param {.    s|
00000060  68 61 70 65 20 7b 0a 20  20 20 20 20 20 64 69 6d  |hape {.      dim|
00000070  3a 20 31 0a 20 20 20 20  20 20 64 69 6d 3a 20 31  |: 1.      dim: 1|
00000080  0a 20 20 20 20 20 20 64  69 6d 3a 20 36 30 0a 20  |.      dim: 60. |
00000090  20 20 20 20 20 64 69 6d  3a 20 36 30 0a 20 20 20  |     dim: 60.   |
000000a0  20 7d 0a 20 20 7d 0a 7d  0a 6c 61 79 65 72 20 7b  | }.  }.}.layer {|
000000b0  0a 20 20 6e 61 6d 65 3a  20 22 63 6f 6e 76 31 22  |.  name: "conv1"|
000000c0  0a 20 20 74 79 70 65 3a  20 22 43 6f 6e 76 6f 6c  |.  type: "Convol|
000000d0  75 74 69 6f 6e 22 0a 20  20 62 6f 74 74 6f 6d 3a  |ution".  bottom:|
000000e0  20 22 64 61 74 61 22 0a  20 20 74 6f 70 3a 20 22  | "data".  top: "|
000000f0  63 6f 6e 76 31 22 0a 20  20 70 61 72 61 6d 20 7b  |conv1".  param {|
00000100  0a 20 20 20 20 6c 72 5f  6d 75 6c 74 3a 20 30 0a  |.    lr_mult: 0.|
00000110  20 20 20 20 64 65 63 61  79 5f 6d 75 6c 74 3a 20  |    decay_mult: |
00000120  31 0a 20 20 7d 0a 20 20  70 61 72 61 6d 20 7b 0a  |1.  }.  param {.|
00000130  20 20 20 20 6c 72 5f 6d  75 6c 74 3a 20 30 0a 20  |    lr_mult: 0. |
00000140  20 20 20 64 65 63 61 79  5f 6d 75 6c 74 3a 20 30  |   decay_mult: 0|
00000150  0a 20 20 7d 0a 20 20 63  6f 6e 76 6f 6c 75 74 69  |.  }.  convoluti|
00000160  6f 6e 5f 70 61 72 61 6d  20 7b 0a 20 20 20 20 6e  |on_param {.    n|
```

How to Locate Model Files?

- Popular model loading locations
  - *Init( ), create_handle( ), create( )*
  - check all *fopen( )* calls

- Names related to DL models
  - keywords: *conv, sigmoid, layer, CAFFEE*
  - functions: *Forward( )*
  - other clues from framework:
    - *Net::NetParameter, NetDef::NetDef*

Reverse Engineering Example #2（APP-B）



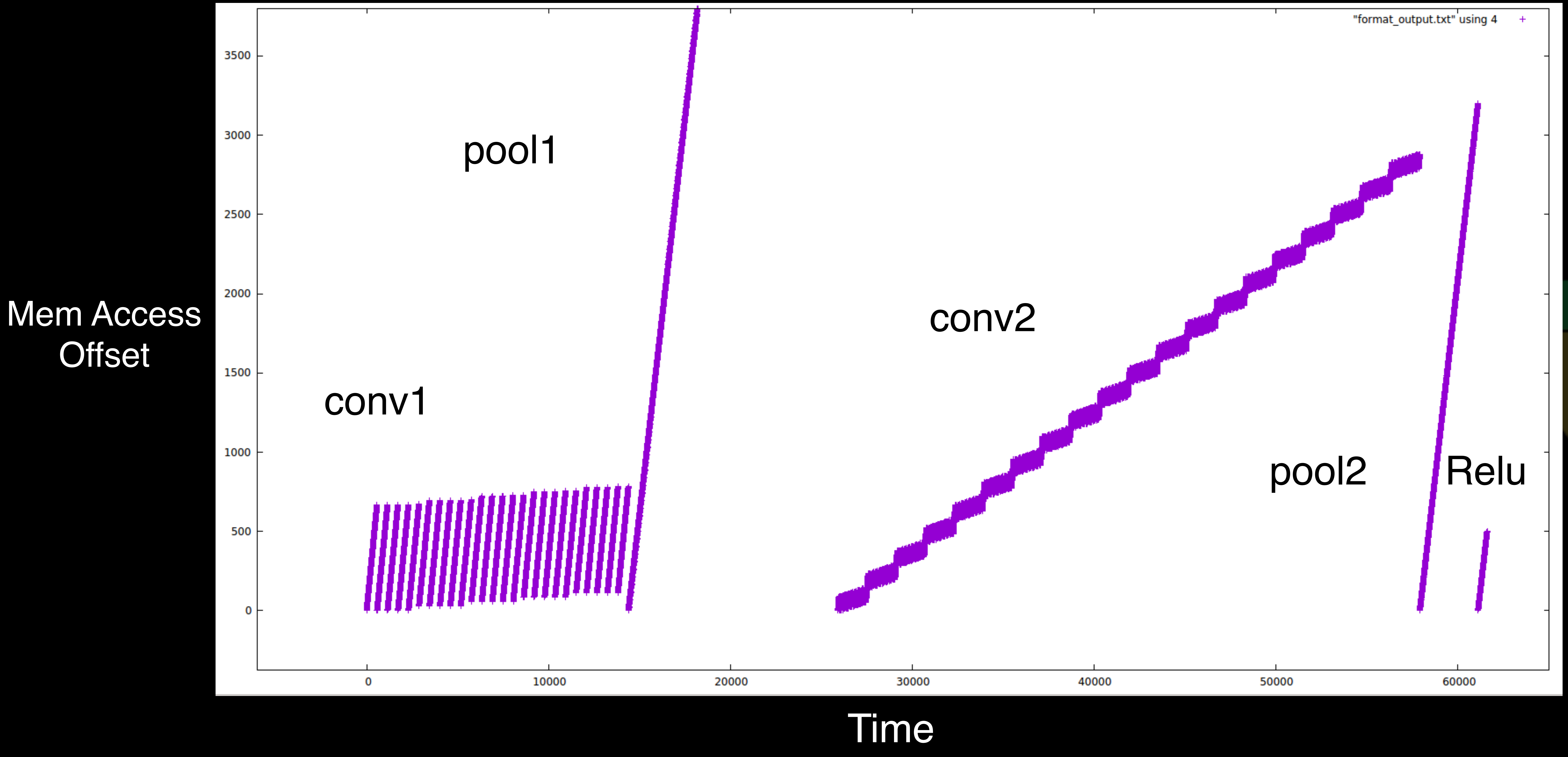
Other Clues to Locate Model Related Functions

- Often framework functions are hidden or in virtual functions
  - check *specific math functions,* and trace back based on cross-references
    - how often does your APP use *exp( )* function family?

$$A = \frac{1}{1+e^{-x}}$$

- For strong obfuscations, dynamic trace and analysis might be needed

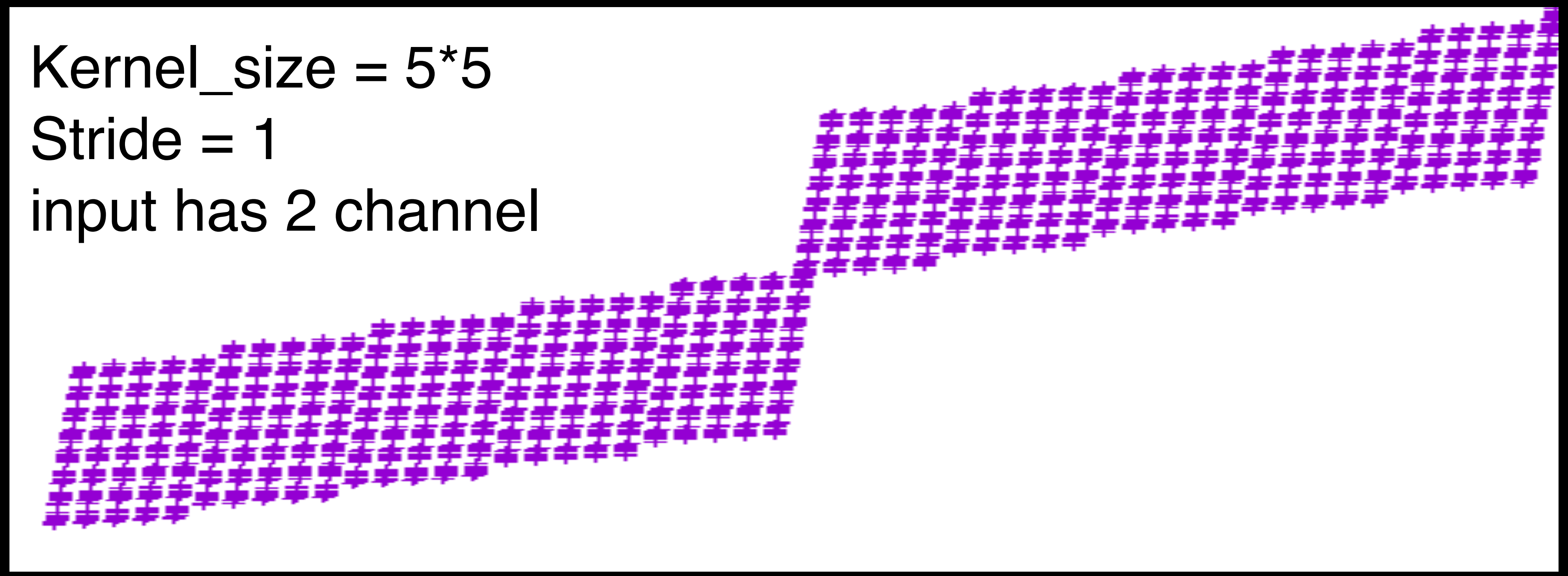
Locate DL Functions by Dynamic Analysis



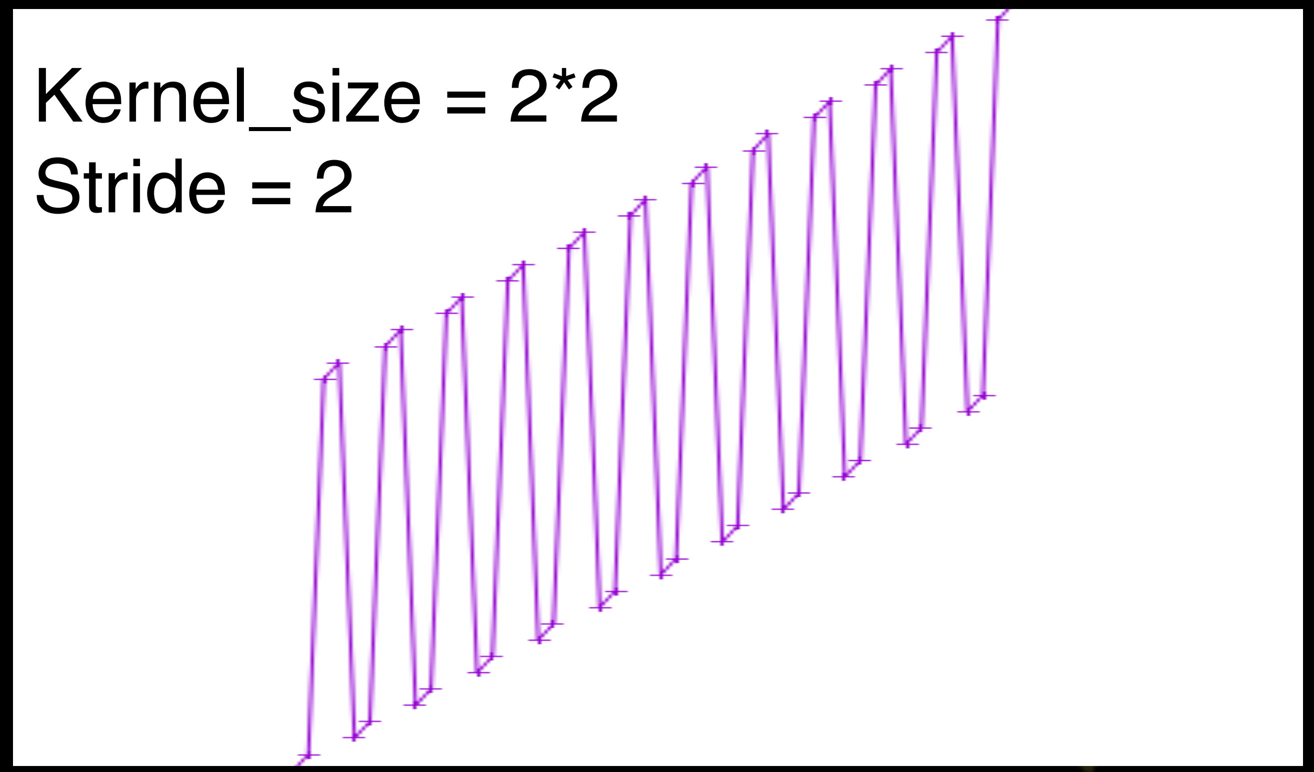
Infer functions through its memory access patterns

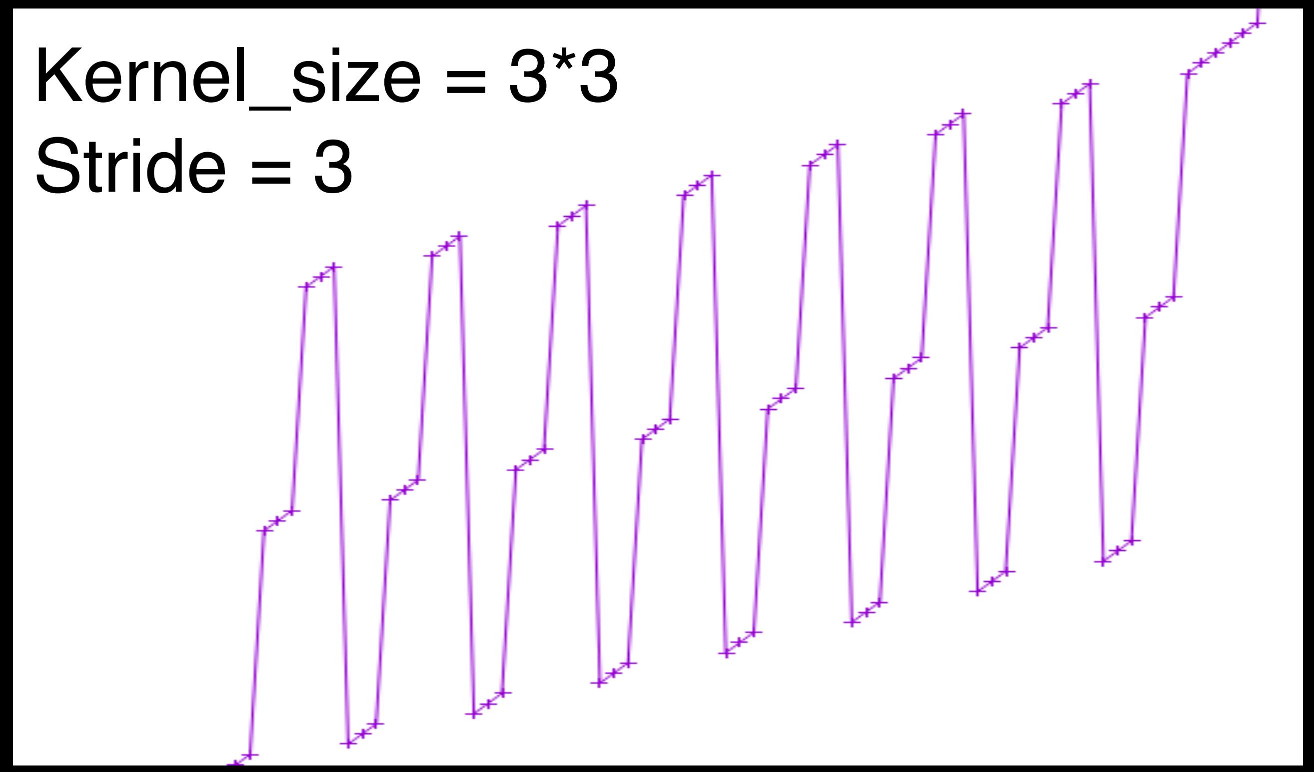
Locate DL Functions by Dynamic Analysis

- Convolution Layer



- Pooling Layer





Layer parameters can be inferred through memory access pattern

Model Reuse Example

- A popular image processing application
- **Built-In App Models**：
  - APK assets directory contains 10 local model files
  - Version with AI hardware: no encryption
  - Version without AI hardware: encryption + steganography
- **Downloaded Models**：
  - For other specific functions, App pulls models from network before their first-time use
  - Models are then saved locally

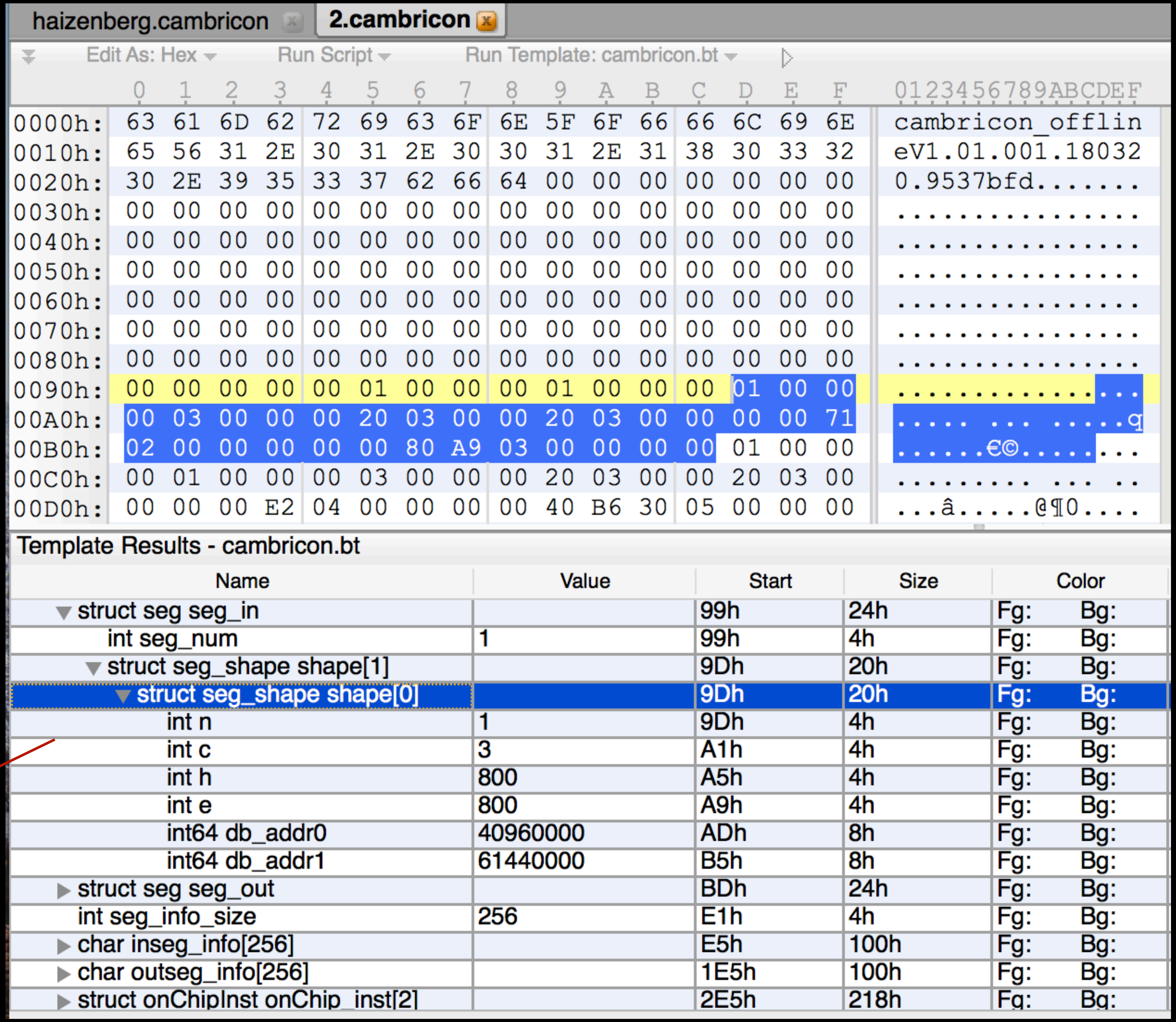
To Use Reverse Engineered Models (Step 1)

- Load model in the new App (here we use the HIAI demo)

```
private class loadModelTask extends AsyncTask<Void, Void, Integer> {
    @Override
    protected Integer doInBackground(Void... voids) {

        int ret = ModelManager.loadModelSync("2", mgr); //load model in file “2.cambricon”
        return ret;
    }
```

To Use Reverse Engineered Models (Step 2)

- Extract the dimensions of input and output layers



Reversed File Format for Cambricon Models

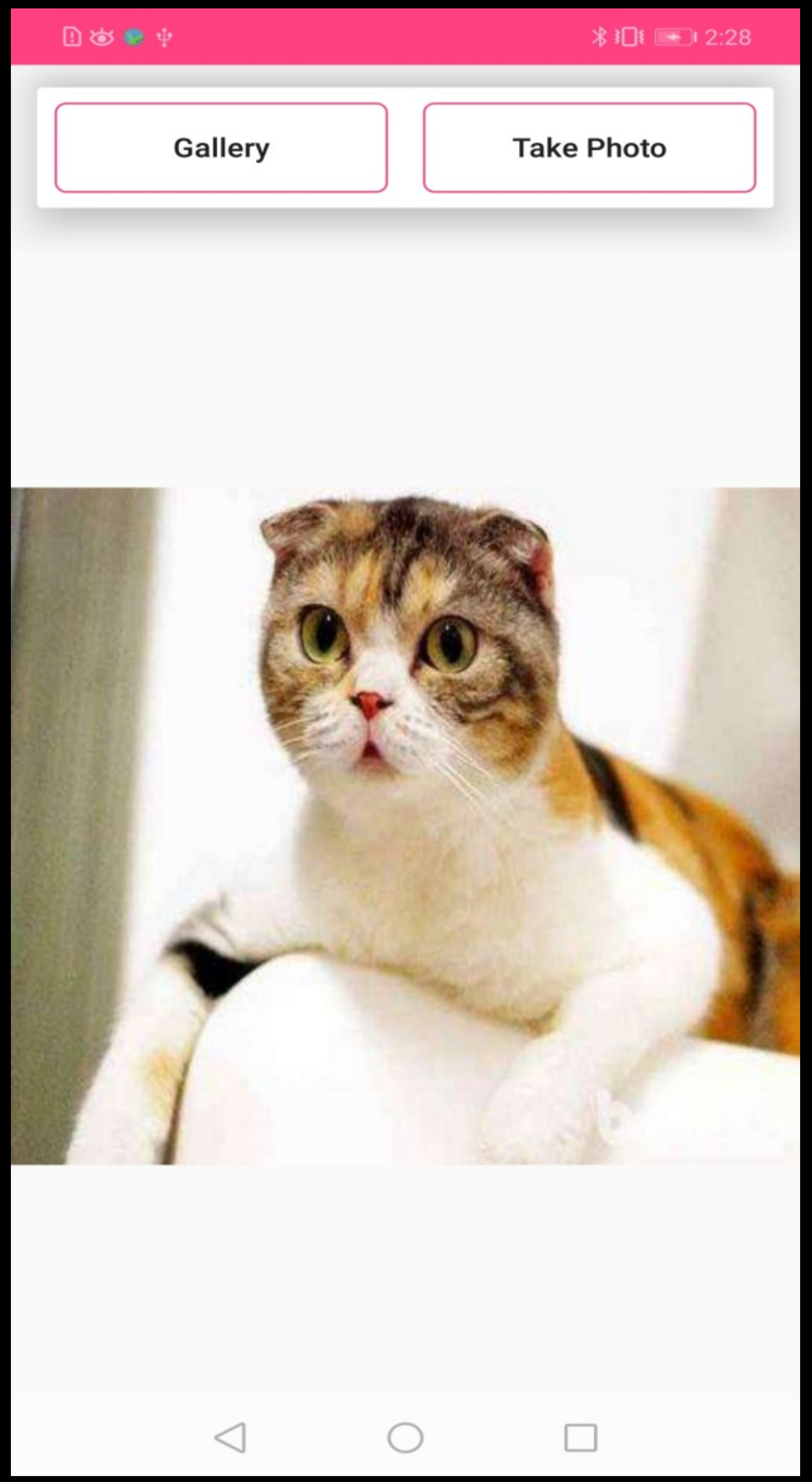
To Use Reverse Engineered Models (Step 3)

- Adjust the input and output scales

```
//inputtensor = HIAI_TensorBuffer_create(1, 3, 299, 299);
inputtensor = HIAI_TensorBuffer_create(1, 3, 800, 800);

HIAI_TensorBuffer *inputtensorbuffer[] = {inputtensor};

//outputtensor = HIAI_TensorBuffer_create(1, 1001, 1, 1);
outputtensor = HIAI_TensorBuffer_create(1, 3, 800, 800);

… …

float *outputBuffer = (float *) HIAI_TensorBuffer_getRawBuffer(outputtensor);

jfloatArray result;
result = (jfloatArray)env->NewFloatArray(3 * 800 * 800);

env->SetFloatArrayRegion(result, 0, 3 * 800 * 800, outputBuffer);
```

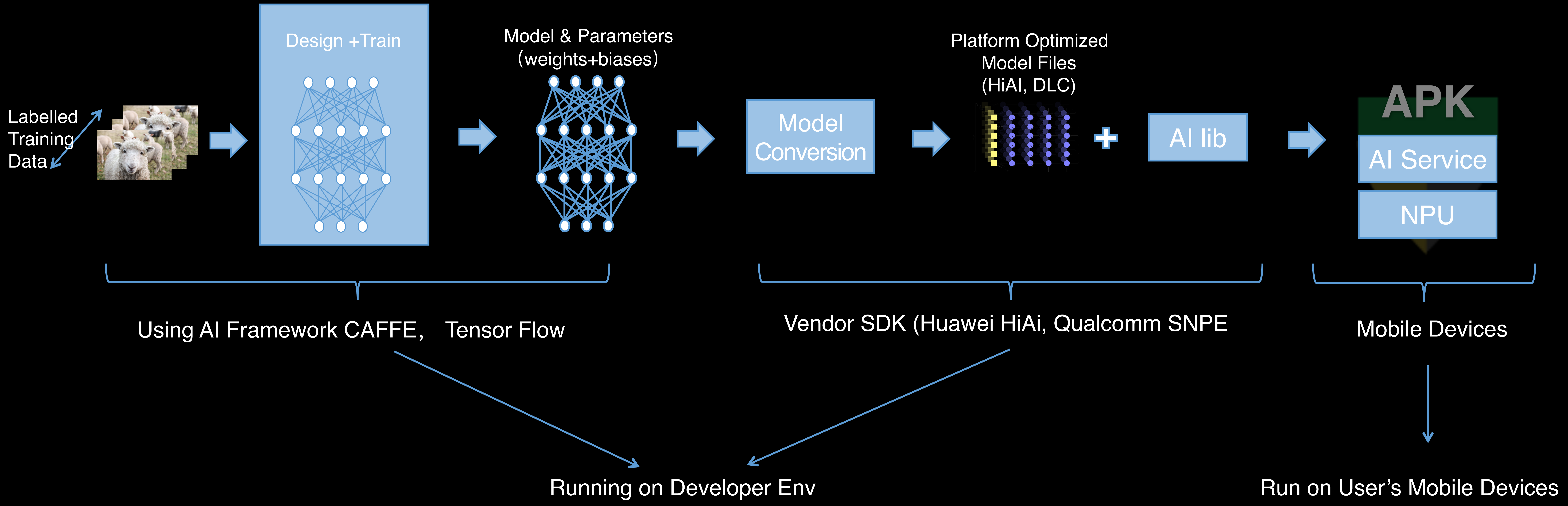
Model “Reuse” Demo

- Porting a model to an open source demo application (no training efforts needed)



New Trend: NPU Protection

Dev Process of AI Mobile App



Example of SNPE Dev Process

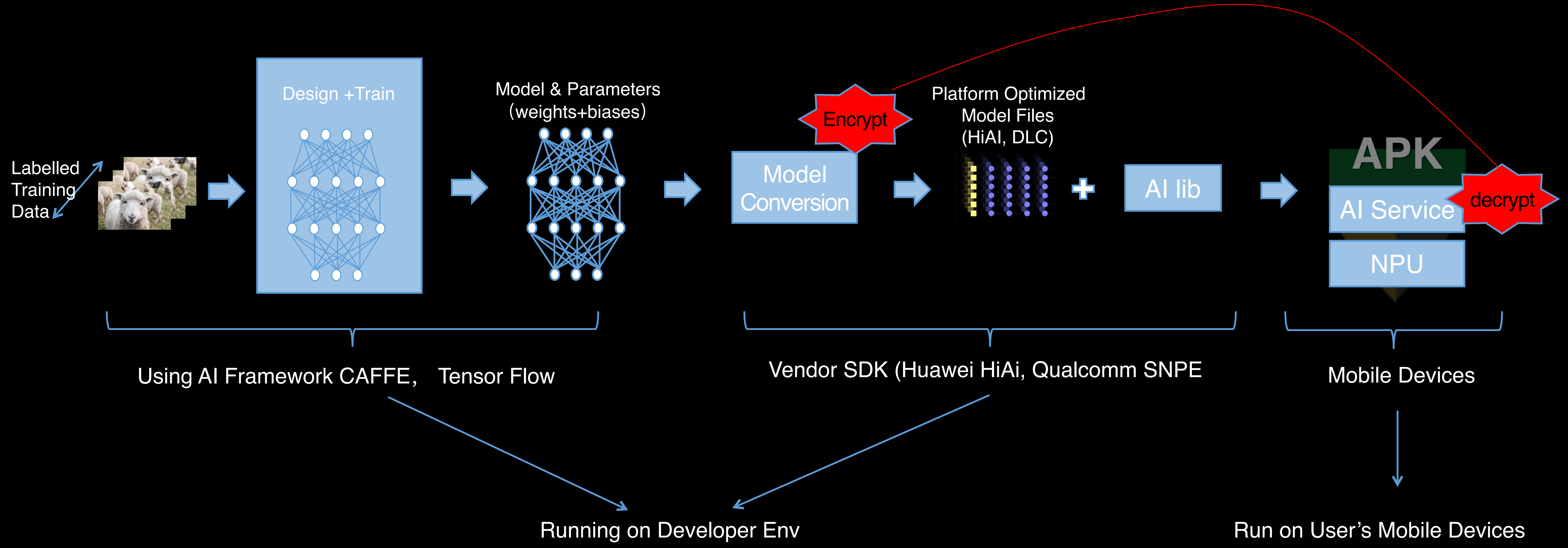
- Model Conversion from Caffee to DLC

```
% snpe-caffe-to-dlc  --caffe_txt $SNPE_ROOT/models/alexnet/caffe/deploy.prototxt \
  --caffe_bin $SNPE_ROOT/models/alexnet/caffe/bvlc_alexnet.caffemodel --dlc alexnet.dlc
```

- We developed a tool to convert DLC back to Caffee model

```
% python dlc-to-caffe.py -i dlcfile -o textmodel
% ./dump_t2m  textmodel   caffemodel_file
```

Dev Process of AI Mobile App



The Current NPU Protection is Weak.

Its current "encryption" can be easily reversed **(details omitted).**

A real encryption + NPU HW decryption can be much stronger.

What about Cloud AI Services?

Threat related to AI Models

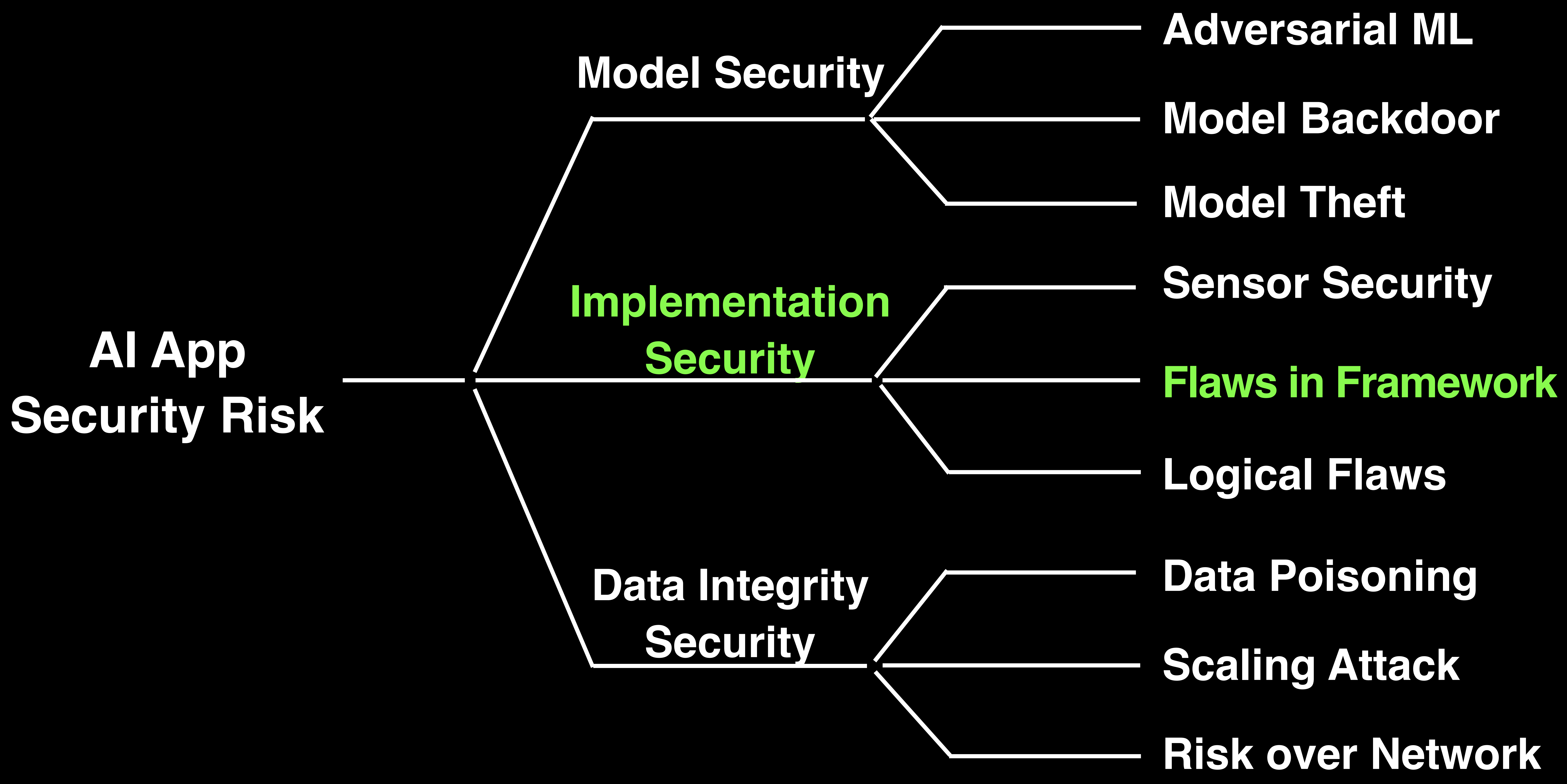
Theory

- model inference attack
- model inverse attack
- membership inference attack

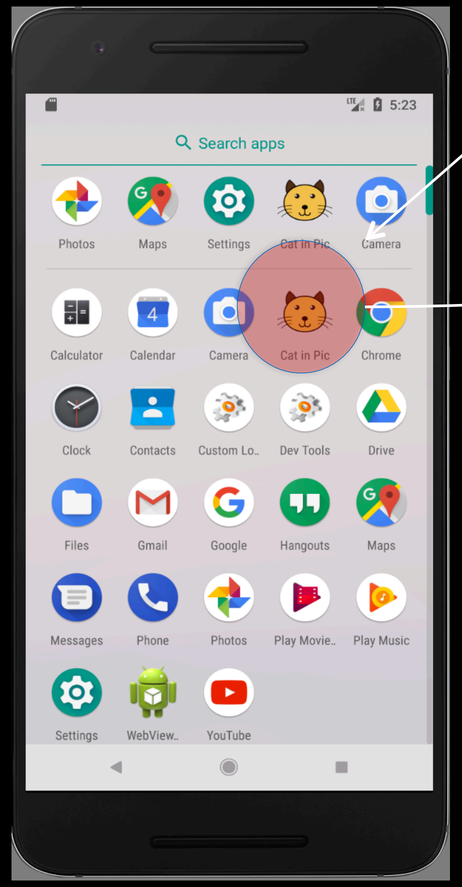
Reality

- reverse engineering attack
- software vulnerability

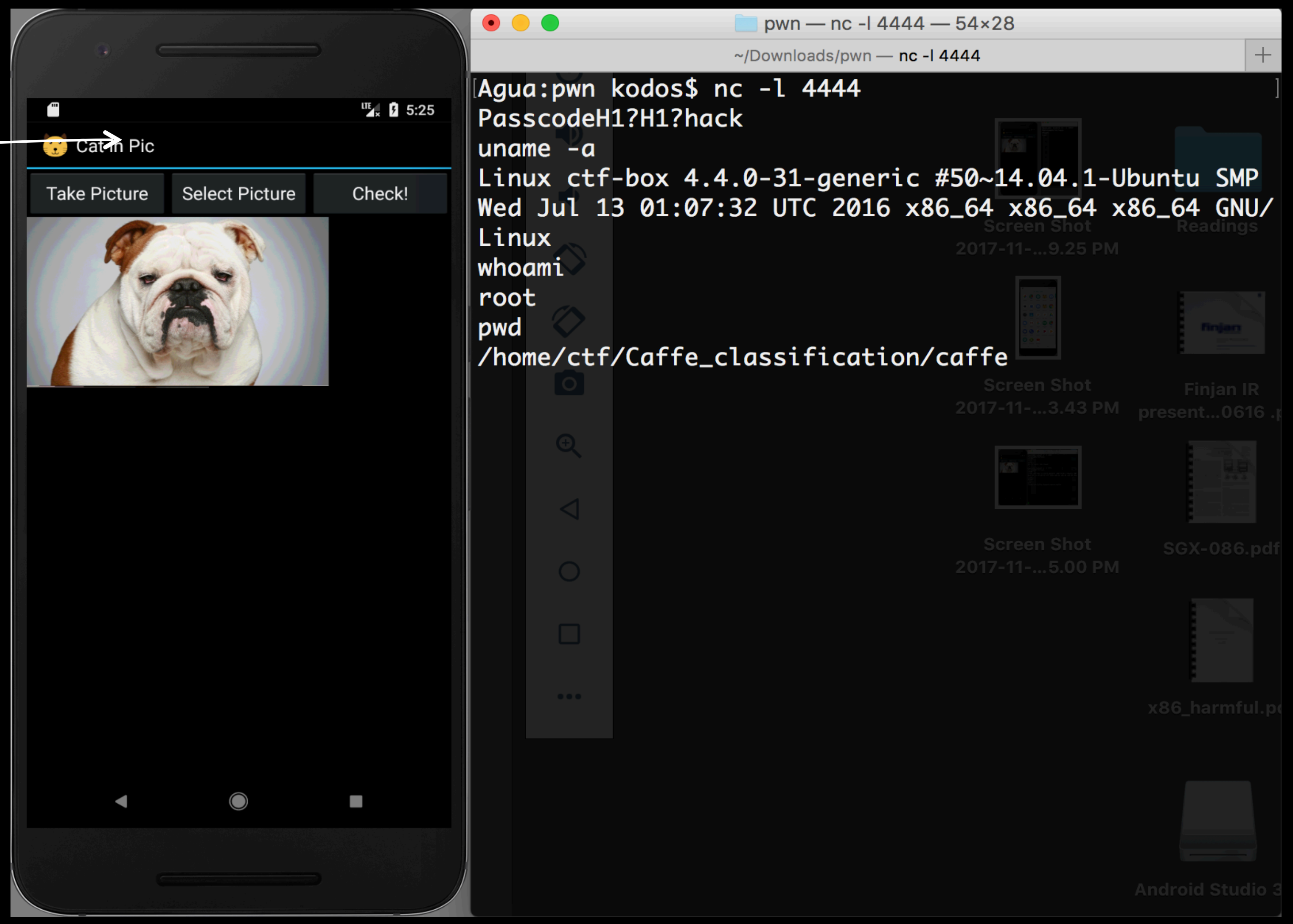
My Prior Work



CatInPic Deep Learning Demo App



upload a crafted image (carried shellcode)



Cloud service connects back, provides a remote shell

CVE 2017-12603 : Heap Overflow

```
/****************** BMP decoder ********************/

bool  BmpDecoder::readHeader()

    if( size >= 36 )
    {
        m_width  = m_strm.getDWord();
        m_height = m_strm.getDWord();
        m_bpp    = m_strm.getDWord() >> 16;
        m_rle_code = (BmpCompression)m_strm.getDWord();
        m_strm.skip(12);
        int clrused = m_strm.getDWord();
        m_strm.skip( size - 36 );

        if( m_width > 0 && m_height != 0 && ……
          (m_bpp == 8 && m_rle_code == BMP_RLE8)))
        {
            iscolor = true;
            result = true;

            if( m_bpp <= 8 )
            {
                memset(m_palette, 0, sizeof(m_palette));
                m_strm.getBytes(m_palette,
                  (clrused == 0? 1<<m_bpp : clrused)*4 );
                  iscolor = IsColorPalette( m_palette, m_bpp );
            }
            else if  …
```

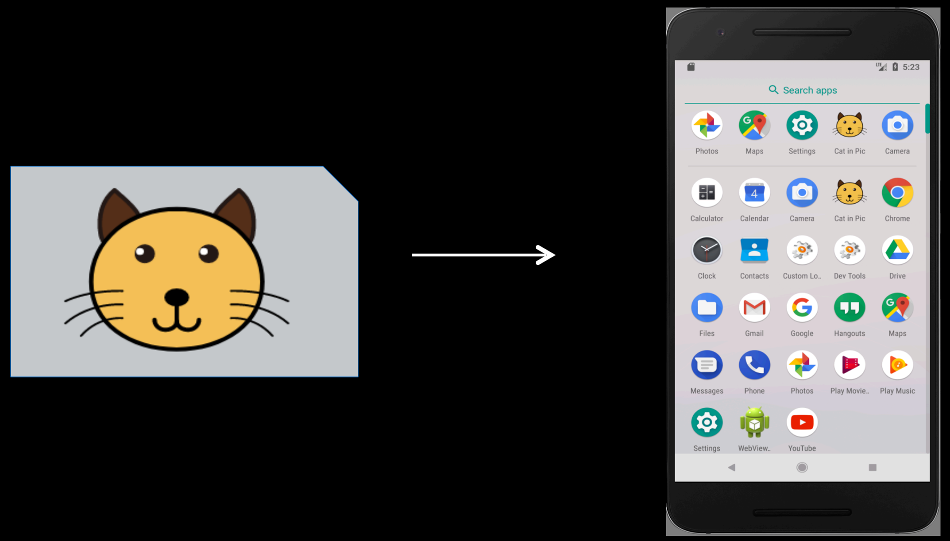
controlled by external input

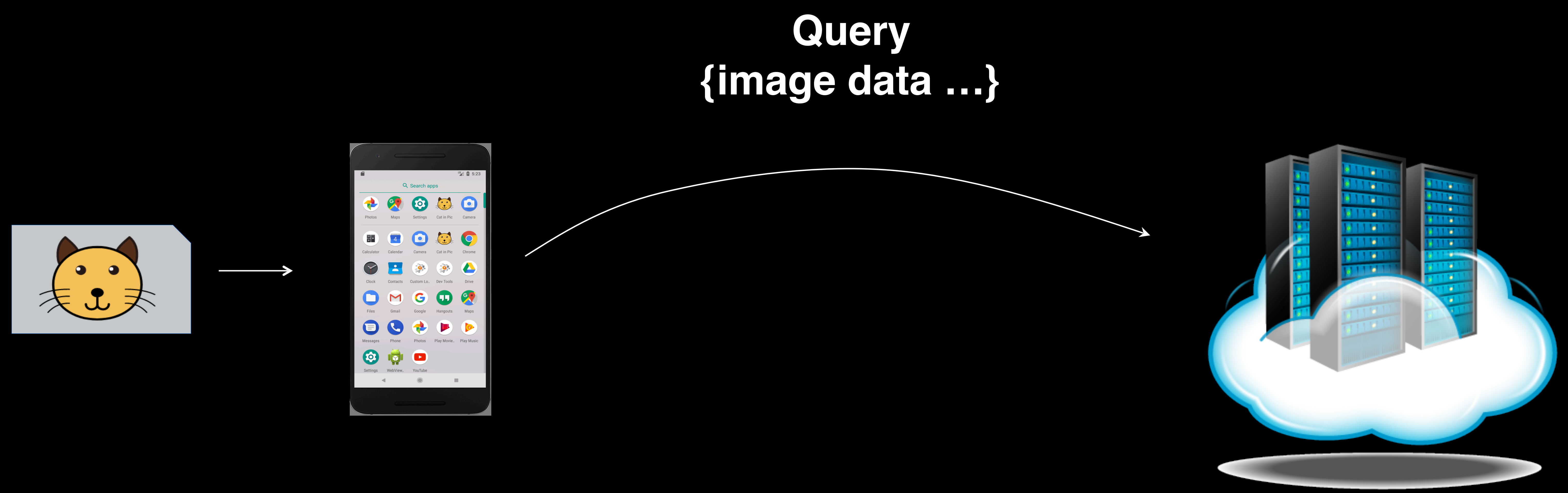
grfmt_bmp.cpp

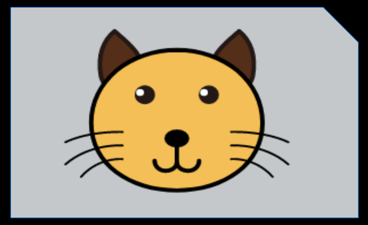
```
int RLByteStream::getBytes( void* buffer, int count )
{
    uchar*  data = (uchar*)buffer;
    int readed = 0;
    assert( count >= 0 );

    while( count > 0 )
    {
        int l;

        for(;;)
        {
            l = (int)(m_end - m_current);
            if( l > count ) l = count;
            if( l > 0 ) break;
            readBlock();
        }
        memcpy( data, m_current, l );
        m_current += l;
        data += l;
        count -= l;
        readed += l;
    }
    return readed;
}
```

heap overflow

bit_strm.cpp




Query
{image data …}

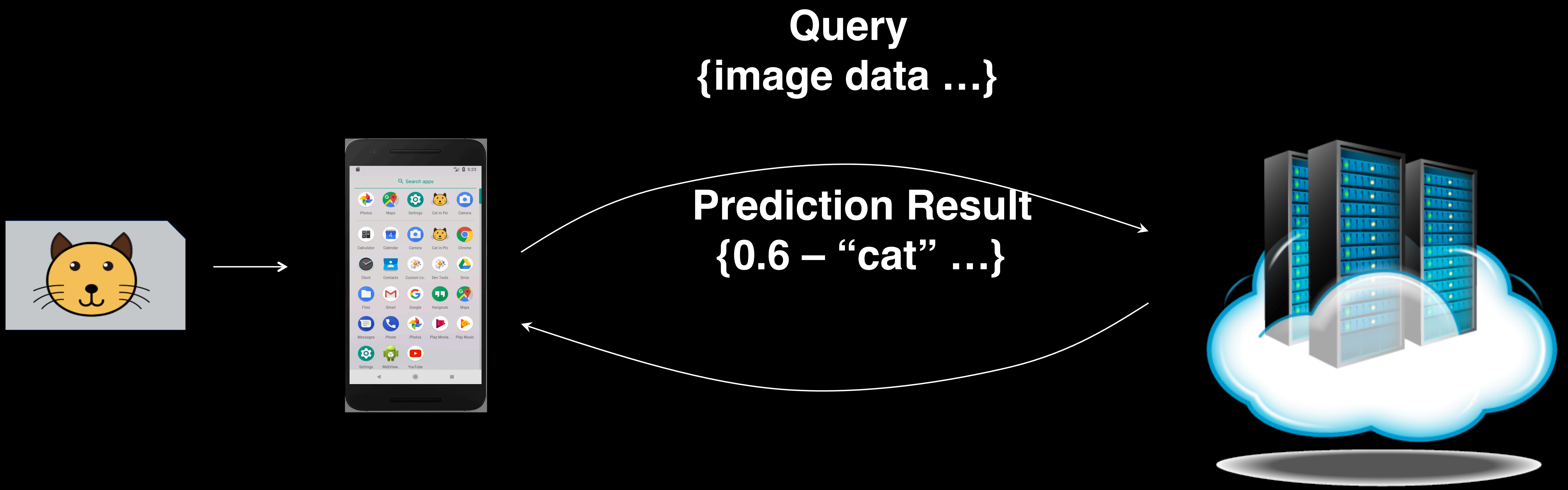


Query
{image data …}

Software at the Cloud side

- **Caffe**
- **CPPClassification**
- **Model: BAIR/BVLC CaffeNet Model**

Downloaded from GitHub on： Oct 25, 2017



Software at the Cloud side

- **Caffe**
- **CPPClassification**
- **Model: BAIR/BVLC CaffeNet Model**

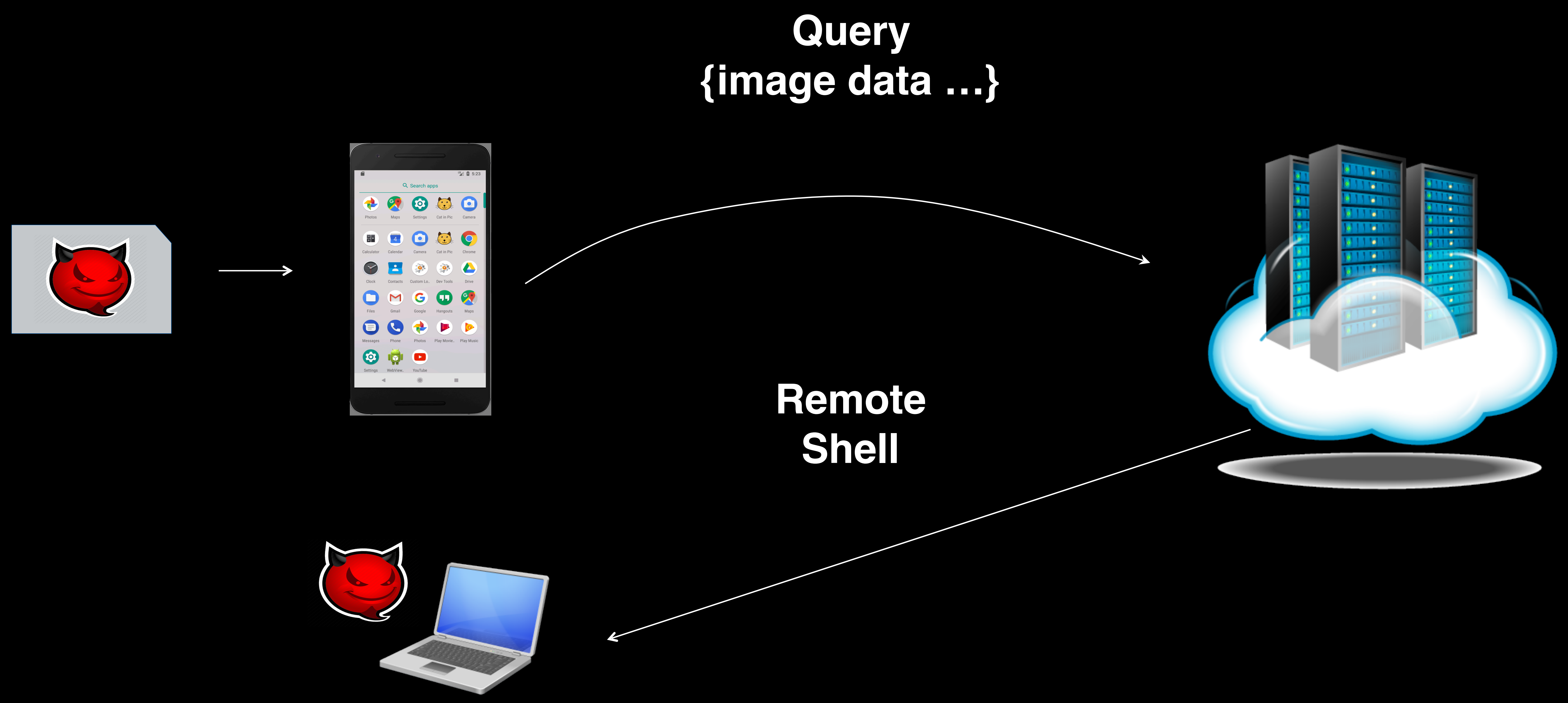
Downloaded from GitHub on： Oct 25, 2017

Query
{image data ...}

Software at the Cloud side

- **Caffe**
- **CPPClassification**
- **Model: BAIR/BVLC CaffeNet Model**

Downloaded from GitHub on： Oct 25, 2017



Software at the Cloud side

- **Caffe**
- **CPPClassification**
- **Model: BAIR/BVLC CaffeNet Model**

Downloaded from GitHub on： Oct 25, 2017

DL Image Classifier Server Setup

```
# Steps to Build Caffe Deep Learning Image Classifier
# Instruction: http://caffe.berkeleyvision.org/installation.html
# Dependencies: OpenCV, OpenBlas

# OpenCV (latest stable version as of Sep 28, 2017)
# https://github.com/opencv/opencv/archive/2.4.13.4.zip
cmake -DCMAKE_BUILD_TYPE=RelWithDebInfo ..
sudo make install

# Openblas
git clone https://github.com/xianyi/OpenBLAS.git
make; sudo install

# Caffe
git clone https://github.com/BVLC/caffe.git
sudo apt-get install libprotobuf-dev libleveldb-dev libsnappy-dev libopencv-dev
libhdf5-serial-dev protobuf-compiler
sudo apt-get install --no-install-recommends libboost-all-dev
sudo apt-get install libgflags-dev libgoogle-glog-dev liblmdb-dev
make all

# Web Service based on Django
pip install -U Django (1.10, 1.11 has been tested)
python manage.py runserver 0.0.0.0:8000
```

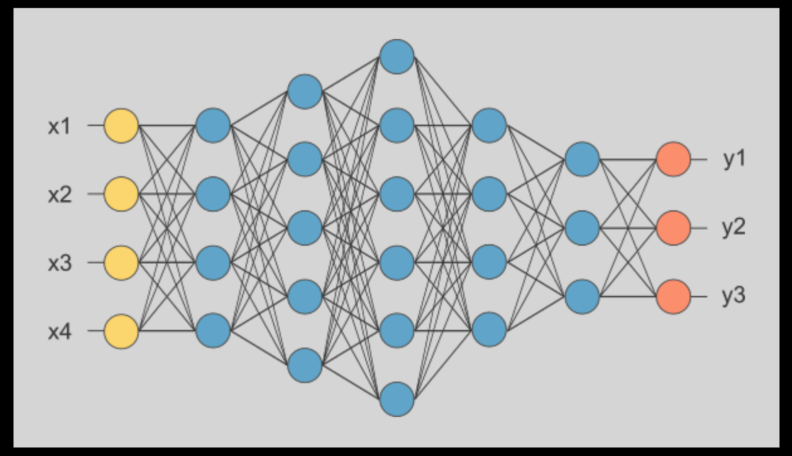
Model Info
Name: BAIR/BVLC CaffeNet Model
Caffemodel:
bvlc_reference_caffenet.caffemodel
Caffemodel_url: http://dl.caffe.berkeleyvision.org/bvlc_reference_caffenet.caffemodel
Caffe_commit:
709dc15af4a06bebda027c1eb2b3f3e3375d5077

AI Application Threats Landscape (Kang's view)

- AI App Security Risk
  - Model Security
    - Adversarial ML
    - Model Backdoor
    - Model Theft
  - Implementation Security
    - Sensor Security
    - Flaws in Framework
    - Logical Flaws
  - Data Integrity Security
    - Data Poisoning
    - Scaling Attack
    - Risk over Network

Summary

- Models and parameters are valuable AI assets
- Endpoint HW capability makes local models increasingly popular
- Local models are vulnerable to reverse engineering
- DL frameworks present strong clues to defeat simple obfuscations
- Software vulnerabilities pose threats to cloud models



Q&A

kangli.ctf@gmail.com