In [20]:
```
os.chdir('/tf/five-video-classification-methods')

from subprocess import call
import os
from extractor import Extractor
from keras.models import load_model
from keras.preprocessing.image import img_to_array, load_img
import numpy as np
from data import DataSet
import numpy as np
from natsort import realsorted, ns
import natsort
```

In [2]:
```
#Importing CNN Model

extractor_model = Extractor()
```

```
WARNING:tensorflow:From /usr/local/lib/python3.5/dist-packages/tensorflow/pytho
n/framework/op_def_library.py:263: colocate_with (from tensorflow.python.framew
ork.ops) is deprecated and will be removed in a future version.
Instructions for updating:
Colocations handled automatically by placer.
```

In [3]:
```
#Importing LSTM Model

model = load_model('data/checkpoints/lstm-features.023-1.096.hdf5')
```

```
WARNING:tensorflow:From /usr/local/lib/python3.5/dist-packages/keras/backend/te
nsorflow_backend.py:3445: calling dropout (from tensorflow.python.ops.nn_ops) w
ith keep_prob is deprecated and will be removed in a future version.
Instructions for updating:
Please use `rate` instead of `keep_prob`. Rate should be set to `rate = 1 - kee
p_prob`.
WARNING:tensorflow:From /usr/local/lib/python3.5/dist-packages/tensorflow/pytho
n/ops/math_ops.py:3066: to_int32 (from tensorflow.python.ops.math_ops) is depre
cated and will be removed in a future version.
Instructions for updating:
Use tf.cast instead.
```

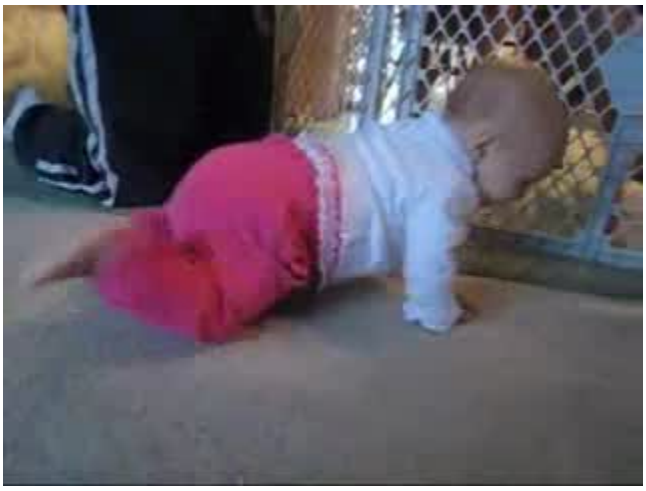
In [48]:
```
# Put you Video PATH in src

src = "crawling.avi"
dest = os.path.join("tests", "frames","%1d.jpg")
call(["ffmpeg", "-i", src,"-vf","fps=10", dest])
```

Out[48]:
```
0
```

In [49]:
```
#sample frame from video
from IPython.display import Image
Image(filename='tests/frames/10.jpg')
```

Out[49]:



In [50]:
```
#Extracting CNN features and saving into numpy array
sequence = []

i = 0
sequence_path = "/tf/five-video-classification-methods/tests/sequence/data_final
os.chdir("/tf/five-video-classification-methods/tests/frames")
sequence = []

files = [f for f in os.listdir('.') if os.path.isfile(f)]

sorted_files = natsort.natsorted(files,reverse=False)

for f in sorted_files:
    i = i + 1
    features = extractor_model.extract(f)
    sequence.append(features)
    if i==40:
        break


np.save(sequence_path, sequence)

len(sequence)
```

Out[50]:
```
40
```

In [52]:
```
#loading numpy array into memory

sequences = np.load("/tf/five-video-classification-methods/tests/sequence/data_f

sequences.shape
```

Out[52]:
```
(40, 2048)
```

In [53]:
```
#Predicting

os.chdir('/tf/five-video-classification-methods')
prediction = model.predict(np.expand_dims(sequences, axis=0))
```

In [54]:
```
#Importing Classnames

os.chdir('/tf/five-video-classification-methods')

from data import DataSet
data = DataSet(seq_length=40, class_limit=4)


def get_classes(self):
        """Extract the classes from our data. If we want to limit them,
        only return the classes we need."""
        classes = []
        for item in self.data:
            if item[1] not in classes:
                classes.append(item[1])

        # Sort them.
        classes = sorted(classes)

        # Return.
        if self.class_limit is not None:
            return classes[:self.class_limit]
        else:
            return classes
```

In [55]:
```
#printing classnames

data.print_class_from_prediction(np.squeeze(prediction, axis=0))
```

```
BabyCrawling: 0.99
Archery: 0.00
ApplyEyeMakeup: 0.00
ApplyLipstick: 0.00
```