Instance	Number of Descriptions		
arch/arch_1	85	eggplant/eggplant_1	82
arch/arch_2	86	eggplant/eggplant_2	82
arch/arch_3	85	eggplant/eggplant_3	86
arch/arch_4	86	eggplant/eggplant_4	85
banana/banana_1	85	lemon/lemon_1	77
banana/banana_2	87	lemon/lemon_2	82
banana/banana_3	85	lemon/lemon_3	84
banana/banana_4	86	lemon/lemon_4	87
cabbage/cabbage_1	85	lime/lime_1	77
cabbage/cabbage_2	87	lime/lime_2	82
cabbage/cabbage_3	85	lime/lime_3	83
cabbage/cabbage_4	86	lime/lime_4	86
carrot/carrot_1	84	orange/orange_1	81
carrot/carrot_2	85	orange/orange_2	83
carrot/carrot_3	84	orange/orange_3	84
carrot/carrot_4	84	orange/orange_4	86
corn/corn_1	83	plum/plum_1	80
corn/corn_2	85	plum/plum_2	83
corn/corn_3	84	plum/plum_3	83
corn/corn_4	87	plum/plum_4	84
cube/cube_1	83	potato/potato_1	81
cube/cube_2	82	potato/potato_2	83
cube/cube_3	84	potato/potato_3	81

cube/cube_4	85	potato/potato_4	84
cuboid/cuboid_1	84	semicylinder/semicylinder_1	81
cuboid/cuboid_2	87	semicylinder/semicylinder_2	83
cuboid/cuboid_3	86	semicylinder/semicylinder_3	80
cuboid/cuboid_4	87	semicylinder/semicylinder_4	83
cucumber/cucumber_1	83	tomato/tomato_1	86
cucumber/cucumber_2	85	tomato/tomato_2	77
cucumber/cucumber_3	86	tomato/tomato_3	84
cucumber/cucumber_4	87	tomato/tomato_4	86
cylinder/cylinder_1	83	triangle/triangle_1	86
cylinder/cylinder_2	86	triangle/triangle_2	78
cylinder/cylinder_3	85	triangle/triangle_3	83
cylinder/cylinder_4	87	triangle/triangle_4	86

Words with TF-IDF value above 50.0*[* ***IDF =*** *math.log(72/count in all documents,* ***TF =*** *count in one doc***)**
['cylinder', 'apple', 'yellow', 'carrot', 'lime', 'blue', 'lemon', 'purple', 'orange', 'banana', 'red', 'cube', 'triangle', 'corn', 'triangular', 'cucumber', 'half', 'cabbage', 'ear', 'tomato', 'potato', 'rectangular', 'cob', 'green', 'eggplant']

> *rgbWords = ['yellow','blue','purple', 'orange','red','green']*
> *shapeWords = ['cylinder','cube', 'triangle','triangular','rectangular']*
> *objWords = ['cylinder', 'apple','carrot', 'lime','lemon','orange', 'banana','cube', 'triangle', 'corn','cucumber', 'cabbage', 'tomato', 'potato', 'cob','eggplant']*

Words with TF-IDF value above 30.0
Above + ['wedge', 'head', 'square', 'block', 'roma', 'circle', 'arch', 'rectangle']

DataSet: 18 categories and 4 instances = 72 instances
Every instance has 3-10 images
One description is for one instance, which means one description is for 3-10 images.

Train - Test Split

75-25% split

Select one instance from every category for testing.
That means there will be **18 instances for testing** and 54 instances for training.
18 instances -> around 60-75 images for testing

Test Verification

Input: Test instance Names,
test image names,
tokens,
probability score matrix,
extracted meaningful tokens for testing
Category for testing(color, shape or object)

PREPROCESSING

Extract meaningful tokens

Now: one of:

rgbWords = ["yellow","blue", "purple","orange", "white", "red", "green"]

shapeWords = ["wedge", "cylinder", "square", "curved", "archshaped","cuboid", "semicylinder", "cube", "triangle", "semicircle", "cylindrical", "triangular","arch", "circle", "rectangular", "rectangle"]

objWords = ["cylinder", "carrot", "tomato", "lime", "cuboid", "prism", "orange", "plantain", "semicylinder", "banana", "cube", "triangle", "corn", "cucumber", "brinjal", "lemon", "cabbage", "arch", "plum", "eggplant"]

For Active Learning:Using TF-IDF for selecting the most important words.

rgbWords = ['yellow','blue','purple', 'orange','red','green']

shapeWords = ['cylinder','cube', 'triangle','triangular','rectangular']

objWords = ['cylinder', 'apple','carrot', 'lime','lemon','orange', 'banana','cube', 'triangle', 'corn','cucumber', 'cabbage', 'tomato', 'potato', 'cob','eggplant']

Other possibilities:

- Anything that occurs > 5 times
- Terms selected by tf*idf
- Terms selected by POS tagging
- ...what else?

TRAINING OCCURS

TESTING, PHASE 1 (PRE-PROCESSING PREDICTIONS):

```
Run all classifiers over all test images and predict class membership
        1 classifier per token in training data set, minus stopwords (including low frequency words)

## For each token, choose a classifier type to test:
For each classifier type
        For each token
                Decide if token is meaningful for that classifier type
                Now: hardcoded in list above
                Other possibilities:
                    -   Performance on training set
                    -   Web search? POS stuff? etc

TESTING, PHASE 2 (VERIFICATION):
For every classifier type:
    For meaningful tokens
            Select positive instances from test set
            Select negative instances from test set
            Test every token 10 times:
                    Require 4-6 images for testing
                            Select 1-4 images from positive instance set
                            Select 1-2 images from negative instance set
                    If the probability (from phase 1) is above 0.5, it is considered as
                    'PASS'/SELECTED
                    Calculate precision, recall and f1 score
                            Get selected images from the probability matrix
                            We have relevant images, which are positive images
```

Calculate overall precision, recall and f1 score

How to Select positive instance for verification

If an instance is described by a token for more than 4 times, that instance will be considered as positive instance of the token

How to select negative instance for verification

```
 For every token c,
     arrayNegInstance = all the instances which are not described by this token
     For every positive instance of c,
           Find negative instances using Doc2Vec
                   Embed descriptive-documents as paragraph vectors
                   Find cosine distances
                   Choose a negative ,  ⅔ rd of all instances which are farthest from the positive
                   instance
```

```
    arrayNegInstance =  intersection(neg instance, arrayNegInstance)
```

arrayNegInstance - list of negative instances