


```

decoder = MeshedDecoder(len(text_field.vocab), 54, 3, text_field.vocab.stoi['<pad>'])
model = Transformer(text_field.vocab.stoi['<bos>'], encoder, decoder).to(device)

data = torch.load('meshed_memory_transformer.pth')
model.load_state_dict(data['state_dict'])

dict_dataset_test = test_dataset.image_dictionary({'image': image_field, 'text': RawField()})
dict_dataloader_test = DataLoader(dict_dataset_test, batch_size=10, num_workers=0)

pred, true, scores = predict_captions(model, dict_dataloader_test, text_field)

```

Meshed-Memory Transformer Evaluation

Evaluation: 100%|██████████| 1/1 [00:00<00:00, 4.47it/s]

In []:

```

# load and display image with annotations

I = io.imread('http://images.cocodataset.org/val2014/COCO_val2014_000000'+ str(img_id) + '.jpg')
plt.imshow(I); plt.axis('off')
ax = plt.gca()
annIds = coco.getAnnIds(imgIds=img_id, iscrowd=None)
anns = coco.loadAnns(annIds)
coco.showAnns(anns)

```

A woman wearing a net on her head cutting a cake.
A woman cutting a large white sheet cake.
A woman wearing a hair net cutting a large sheet cake.
there is a woman that is cutting a white cake
A woman marking a cake with the back of a chef's knife.



In []:

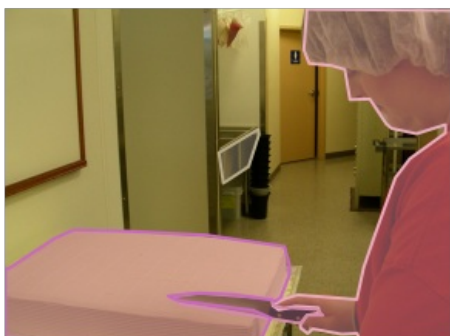
```

# load and display image with instance annotation
annFile = 'annotations/instances_val2014.json'
coco_caps=COCO(annFile)
annIds = coco_caps.getAnnIds(imgIds=img_id);
anns = coco_caps.loadAnns(annIds)

plt.imshow(I); plt.axis('off')
annIds = coco_caps.getAnnIds(imgIds=img_id, iscrowd=None)
anns = coco_caps.loadAnns(annIds)
coco_caps.showAnns(anns)

```

loading annotations into memory...
Done (t=4.80s)
creating index...
index created!



In []:

```
print("Predicted Annotation: {}".format("".join(list(true.values())[0])))  
  
print("\nScores:\n{}".format(scores))
```

Predicted Annotation: a person cutting a large cake with a knife

Scores:

{'BLEU': [0.4448541126327038, 0.2110128333603988, 1.7205604578262437e-06, 5.10607476252634e-09], 'METEOR': 0.17123645094347326, 'ROUGE': 0.3929146537842191, 'CIDEr': 0.0}

Loading [Mathjax]/jax/output/CommonHTML/fonts/TeX/fontdata.js