In[]:

```
## Name : Tanmay Gujar

 ##Rollno: I4159
```

In[1]:

```
importmatplotlib.pyplotasplt
importseabornassns
importmatplotlibasmpl
importmatplotlib.pylabaspylab
importnumpyasnp
%matplotlibinline
```

In[2]:

```
#DataPrepration
importre
```

In[3]:

```
sentences="""Weareabouttostudytheideaofacomputationalprocess.Computation
alprocessesareabstractbeingsthatinhabitcomputers.
Astheyevolve,processesmanipulateotherabstractthingscalleddata.
Theevolutionofaprocessisdirectedbyapatternofrules
calledaprogram.Peoplecreateprogramstodirectprocesses.Ineffect,weconjure
thespiritsofthecomputerwithourspells."""
```

Clean Data

In[4]:

```
#removespecialcharacters
sentences=re.sub('[^A-Za-z0-9]+','',sentences)

#remove1letterwords
sentences=re.sub(r'(?:^|)\w(?:$|)','',sentences).strip()

#lowerallcharacters
sentences=sentences.lower()
```

Vocabulary

In[5]:

```
words=sentences.split()vocab=s
et(words)
```

In[6]:

```
vocab_size=len(vocab)e
mbed_dim=10
context_size=2
```

Implementation

In[7]:

```
word_to_ix={word:ifori,wordinenumerate(vocab)}ix_to_w
ord={i:wordfori,wordinenumerate(vocab)}
```

Data bags

In[8]:

```
#data-[(context),target]

data=[]
foriinrange(2,len(words)-2):
    [words[i-2],words[i-1],words[i+1],words[i+2]]
```

```
    target=words[i]
    data.append((context,target))print(data[:5])
```

```
[(['we','are','to','study'],'about'),(['are','about','study','the'],'to'),(['abo
ut','to','the','idea'],'study'),(['to','study','idea','of'],'the'),(['study','the','of',
'computational'],'idea')]
```

Embeddings

In[9]:

```
embeddings=np.random.random_sample((vocab_size,embed_dim))
```

Linear Model

In[10]:

```
deflinear(m,theta):w
    =theta
    returnm.dot(w)
```

Log softmax + NLLloss = Cross Entropy

In[11]:

```
deflog_softmax(x):
    e_x=np.exp(x-np.max(x))
    returnnp.log(e_x/e_x.sum())
```

In[12]:

```
defNLLLoss(logs,targets):
    out=logs[range(len(targets)),targets]
    return-out.sum()/len(out)
```

In[13]:

```
deflog_softmax_crossentropy_with_logits(logits,target):

    out=np.zeros_like(logits)
    out[np.arange(len(logits)),target]=1

    softmax=np.exp(logits)/np.exp(logits).sum(axis=-1,keepdims=True)

    return(-out+softmax)/logits.shape[0]
```

Forward function

In[14]:

```
defforward(context_idxs,theta):
    m=embeddings[context_idxs].reshape(1,-
    1)n=linear(m,theta)
    o=log_softmax(n)

    returnm,n,o
```

Backward function

In[15]:

```
defbackward(preds,theta,target_idxs):m,
    n,o=preds

    dlog=log_softmax_crossentropy_with_logits(n,target_idxs)dw=m.T
    .dot(dlog)

    returndw
```

Optimize function

In[16]:
```
defoptimize(theta,grad,lr=0.03):th
    eta-=grad*lr
    returntheta
```

Training

In[17]:
```
#Genratetrainingdata

theta=np.random.uniform(-1,1,(2*context_size*embed_dim,vocab_size))
```

In[18]:
```
epoch_losses={}

forepochinrange(80):lo

    sses=[]

    forcontext,targetindata:
        context_idxs=np.array([word_to_ix[w]forwincontext])preds
        =forward(context_idxs,theta)

        target_idxs=np.array([word_to_ix[target]])los
        s=NLLLoss(preds[-1],target_idxs)

        losses.append(loss)

        grad=backward(preds,theta,target_idxs)the
        ta=optimize(theta,grad,lr=0.03)


    epoch_losses[epoch]=losses
```

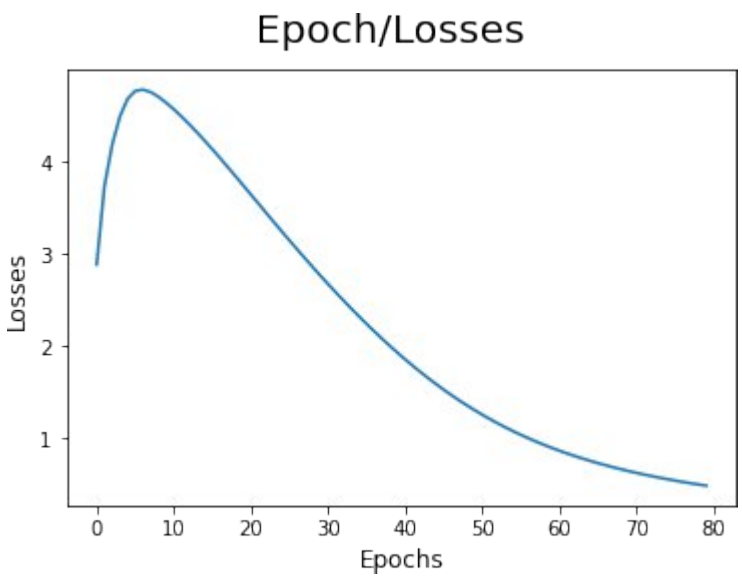
Analyze

Plot loss/epoch

In[19]:
```
ix=np.arange(0,80)

fig=plt.figure()
fig.suptitle('Epoch/Losses',fontsize=20)
plt.plot(ix,[epoch_losses[i][0]foriinix])plt
.xlabel('Epochs',fontsize=12)
plt.ylabel('Losses',fontsize=12)
```

Out[19]:
```
Text(0,0.5,'Losses')
```



Predict function

In[20]:

```
defpredict(words):
    context_idxs=np.array([word_to_ix[w]forwinwords])preds
    =forward(context_idxs,theta)
    word=ix_to_word[np.argmax(preds[-1])]

    returnword
```

In[21]:

```
#(['we','are','to','study'],'about')
predict(['we','are','to','study'])
```

Out[21]: 'about'

Accuracy

In[22]:

```
defaccuracy():w
    rong=0

    forcontext,targetindata:
        if(predict(context)!=target):wr
            ong+=1

    return(1-(wrong/len(data)))
```

In[23]:

```
accuracy()
```

Out[23]: 1.0

In[24]:

```
predict(['processes','manipulate','things','study'])
```

Out[24]: 'effect'

In[]: