Graph Plot for LSTM Models

December 12, 2018

In [ ]:
```
# Depression Analysis in Bangla with LSTM-RNN RESULTS
# copyright (c) ABDUL HASIB UDDIN <abdulhasibuddin@gmail.com>
# LICENSE: GNU General Public License v3.0
```

In [1]:
```
import matplotlib.pyplot as plt
from scipy.interpolate import spline
import numpy as np
```

In [2]:
```
# LSTM MODEL STATISTICS:
```

In [3]:
```
# lstm validation accuracies:
lstm_with_unique_1 = [0.445,0.409,0.327,0.500,0.464,0.445,0.309,0.500,0.427,0.455,0.50
lstm_with_unique_2 = [0.436,0.345,0.273,0.555,0.455,0.436,0.500,0.500,0.436,0.436,0.50
lstm_with_unique_3 = [0.445,0.318,0.309,0.491,0.400,0.418,0.500,0.491,0.391,0.400,0.65
lstm_with_unique_4 = [0.345,0.336,0.355,0.745,0.336,0.364,0.555,0.609,0.336,0.418,0.73
lstm_with_unique_5 = [0.291,0.382,0.473,0.500,0.382,0.355,0.618,0.445,0.482,0.318,0.68
lstm_with_unique_6 = [0.500,0.430,0.390]
lstm_with_unique_7 = [0.450,0.420,0.440,0.440,0.560,0.530,0.550]
lstm_with_unique_8 = [0.435,0.496,0.435,0.400,0.426,0.270,0.522,0.504,0.548,0.461,0.42
lstm_with_unique_9 = [0.475,0.475,0.466,0.517,0.449,0.466,0.466,0.466,0.458,0.398,0.37
lstm_with_unique_10 = [0.410,0.500,0.500,0.470,0.510,0.410,0.450,0.620,0.660,0.480,0.5
lstm_with_unique_11 = [0.430,0.400,0.500,0.480,0.510,0.500,0.430]
lstm_with_unique_12 = [0.391,0.435,0.365,0.357,0.426,0.322,0.504,0.487,0.487,0.487,0.4
lstm_with_unique_13 = [0.478,0.470,0.496,0.452,0.417,0.365,0.504,0.530,0.617,0.478,0.4
lstm_with_unique_14 = [0.357,0.426,0.365,0.357,0.391,0.270,0.539,0.496,0.478,0.443,0.4

print(len(lstm_with_unique_9))
```

```
75
```

In [4]:
```
# iterations (x-axis):

x_axis_1 = []
for iter_no in range(1,470+1):
    if iter_no%25 == 0:
        x_axis_1.append(iter_no)
x_axis_2 = []
```

```
for iter_no in range(1,470+1):
    if iter_no%25 == 0:
        x_axis_2.append(iter_no)
x_axis_3 = []
for iter_no in range(1,470+1):
    if iter_no%25 == 0:
        x_axis_3.append(iter_no)
x_axis_4 = []
for iter_no in range(1,470+1):
    if iter_no%25 == 0:
        x_axis_4.append(iter_no)
x_axis_5 = []
for iter_no in range(1,470+1):
    if iter_no%25 == 0:
        x_axis_5.append(iter_no)
x_axis_6 = []
for iter_no in range(1,90+1):
    if iter_no%25 == 0:
        x_axis_6.append(iter_no)
x_axis_7 = []
for iter_no in range(1,180+1):
    if iter_no%25 == 0:
        x_axis_7.append(iter_no)
x_axis_8 = []
for iter_no in range(1,560+1):
    if iter_no%25 == 0:
        x_axis_8.append(iter_no)
x_axis_9 = []
for iter_no in range(1,1880+1):
    if iter_no%25 == 0:
        x_axis_9.append(iter_no)
x_axis_10 = []
for iter_no in range(1,370+1):
    if iter_no%25 == 0:
        x_axis_10.append(iter_no)
x_axis_11 = []
for iter_no in range(1,185+1):
    if iter_no%25 == 0:
        x_axis_11.append(iter_no)
x_axis_12 = []
for iter_no in range(1,560+1):
    if iter_no%25 == 0:
        x_axis_12.append(iter_no)
x_axis_13 = []
for iter_no in range(1,560+1):
    if iter_no%25 == 0:
        x_axis_13.append(iter_no)
x_axis_14 = []
```

```
        for iter_no in range(1,1880+1):
            if iter_no%25 == 0:
                x_axis_14.append(iter_no)
```

In [5]:
```
x_list = [x_axis_1,x_axis_2,x_axis_3,x_axis_4,x_axis_5,x_axis_6,x_axis_7,x_axis_8,x_axi
model_list = [lstm_with_unique_1,lstm_with_unique_2,lstm_with_unique_3,lstm_with_uniqu
required_iteration_list = [470,470,470,470,470,90,180,560,1880,370,185,560,560,1880]
required_epoch_list = [5,5,5,5,5,5,10,3,2,10,5,3,3,10]
test_acc_list = [50.0,73.6,72.7,69.1,70.9,36.0,59.0,73.9,77.1,57.0,40.0,73.9,42.6,62.6

print(len(x_list))
print(len(model_list))
print(len(required_iteration_list))
print(len(required_epoch_list))
print(len(test_acc_list))
```

```
14
14
14
14
14
```

In [6]:
```
# average validation accuracies:
avg_val_acc_list = []
for model in model_list:
    avg_val_acc_list.append(sum(model)/len(model))
print(len(avg_val_acc_list))
print(avg_val_acc_list)
```

```
14
[0.44738888888888895, 0.4585555555555556, 0.4752222222222222, 0.4777222222222222, 0.4439444444
```

In [7]:
```
best_model_val_acc_1 = lstm_with_unique_9 # 77.1%
best_model_val_loss_1 = [0.282,0.251,0.283,0.250,0.253,0.202,0.254,0.254,0.358,0.189,0
best_model_val_acc_2 = lstm_with_unique_8 # 73.9%
best_model_val_loss_2 = [0.245,0.198,0.221,0.251,0.183,0.268,0.205,0.225,0.257,0.238,0

print(len(best_model_val_acc_1))
print(len(best_model_val_loss_1))
print(len(best_model_val_acc_2))
print(len(best_model_val_loss_2))
```

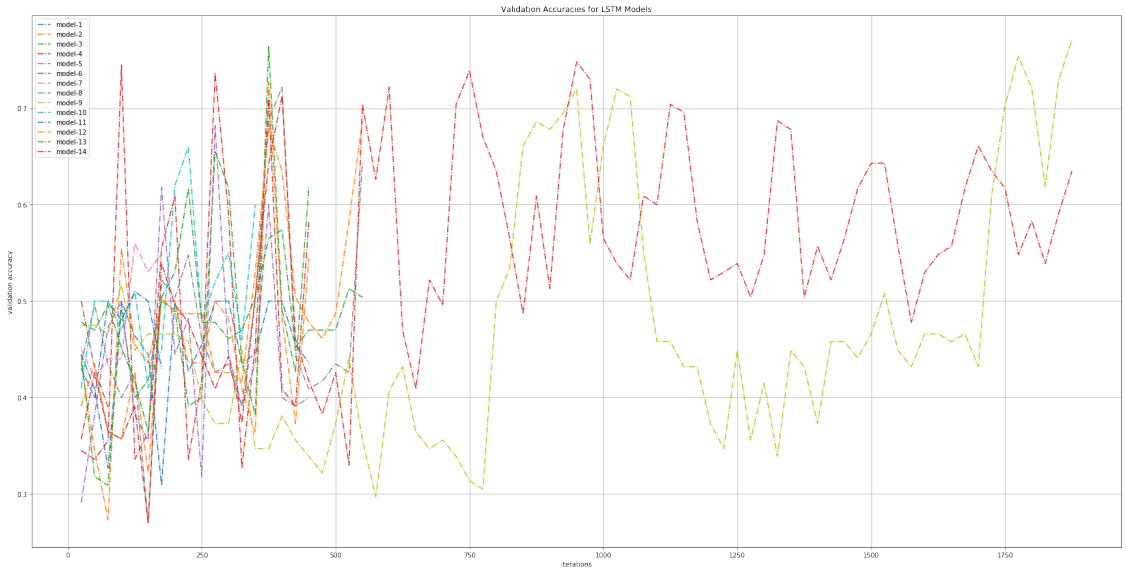
```
75
75
22
22
```

In [ ]:

In [8]:
```
for i in range(0,len(x_list)):
    x_axis_name = 'len(x_axis_'+str(i+1)+') ='
    model_name = '; len(lstm_with_unique_'+str(i+1)+') ='
    x_axis_length = len(x_list[i])
    model_length = len(model_list[i])
    print(x_axis_name,x_axis_length, model_name,model_length, '; status =',x_axis_leng
```

```
len(x_axis_1) = 18 ; len(lstm_with_unique_1) = 18 ; status = True
len(x_axis_2) = 18 ; len(lstm_with_unique_2) = 18 ; status = True
len(x_axis_3) = 18 ; len(lstm_with_unique_3) = 18 ; status = True
len(x_axis_4) = 18 ; len(lstm_with_unique_4) = 18 ; status = True
len(x_axis_5) = 18 ; len(lstm_with_unique_5) = 18 ; status = True
len(x_axis_6) = 3 ; len(lstm_with_unique_6) = 3 ; status = True
len(x_axis_7) = 7 ; len(lstm_with_unique_7) = 7 ; status = True
len(x_axis_8) = 22 ; len(lstm_with_unique_8) = 22 ; status = True
len(x_axis_9) = 75 ; len(lstm_with_unique_9) = 75 ; status = True
len(x_axis_10) = 14 ; len(lstm_with_unique_10) = 14 ; status = True
len(x_axis_11) = 7 ; len(lstm_with_unique_11) = 7 ; status = True
len(x_axis_12) = 22 ; len(lstm_with_unique_12) = 22 ; status = True
len(x_axis_13) = 22 ; len(lstm_with_unique_13) = 22 ; status = True
len(x_axis_14) = 75 ; len(lstm_with_unique_14) = 75 ; status = True
```

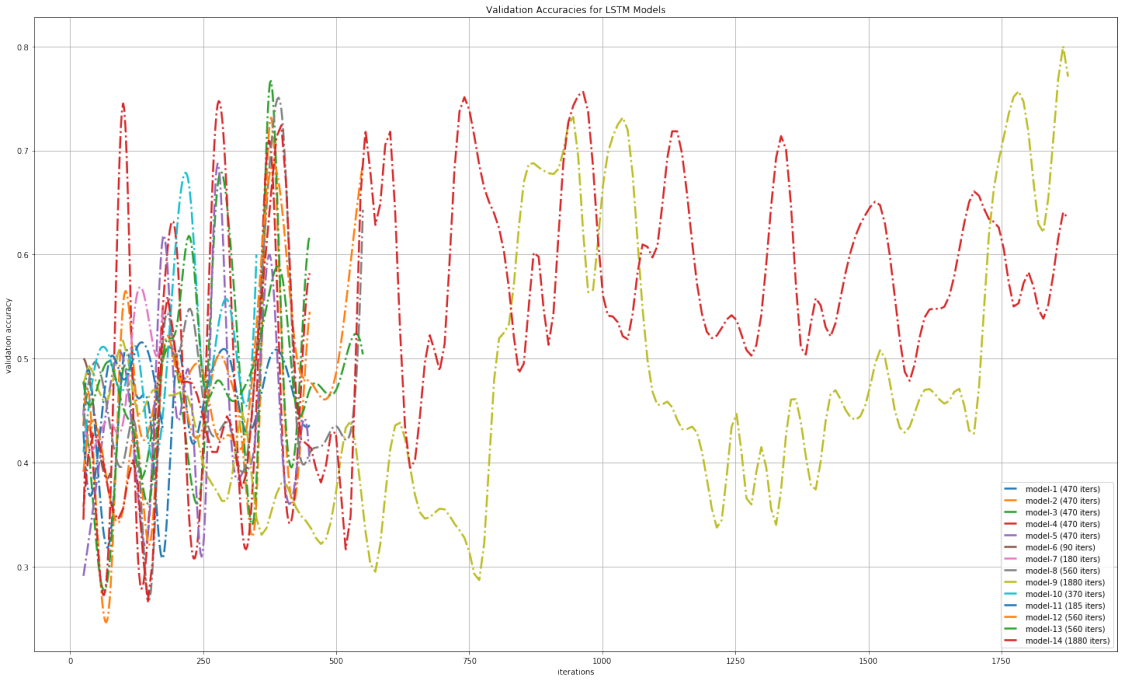
In [9]:
```
# lstm plotting validation accuracies against iterations:

    linestyle='-.'
    linewidth = 2.5

    plt.figure(figsize=(30,15))
    plt.title('Validation Accuracies for LSTM Models')
    plt.xlabel('iterations')
    plt.ylabel('validation accuracy')

    for i in range(0,len(x_list)):
        label = "model-"+str(i+1)
        x = x_list[i]
        y = model_list[i]
        plt.plot(x, y, linestyle=linestyle, label=label)

    plt.grid(True)
    plt.legend()
    plt.show()
```



In [10]:
```
# lstm plotting validation accuracies against iterations:
smoothing_factor = 200
linestyle='-.'
linewidth = 2.5

#plt.figure(figsize=(13,8))
plt.figure(figsize=(25,15))
plt.title('Validation Accuracies for LSTM Models')
plt.xlabel('iterations')
plt.ylabel('validation accuracy')

for i in range(0,len(x_list)):
    label = "model-"+str(i+1)+" ("+str(required_iteration_list[i])+" iters)"
    x = x_list[i]
    y = model_list[i]
    x_sm = np.array(x)
    y_sm = np.array(y)
    x_smooth = np.linspace(x_sm.min(), x_sm.max(), smoothing_factor)
    y_smooth = spline(x, y, x_smooth)
    plt.plot(x_smooth, y_smooth, linestyle=linestyle, linewidth=linewidth, label=label

plt.grid(True)
plt.legend()
plt.savefig('images\lstm_image_1_plotting _validation _accuracies _against _iterations
plt.show()
```

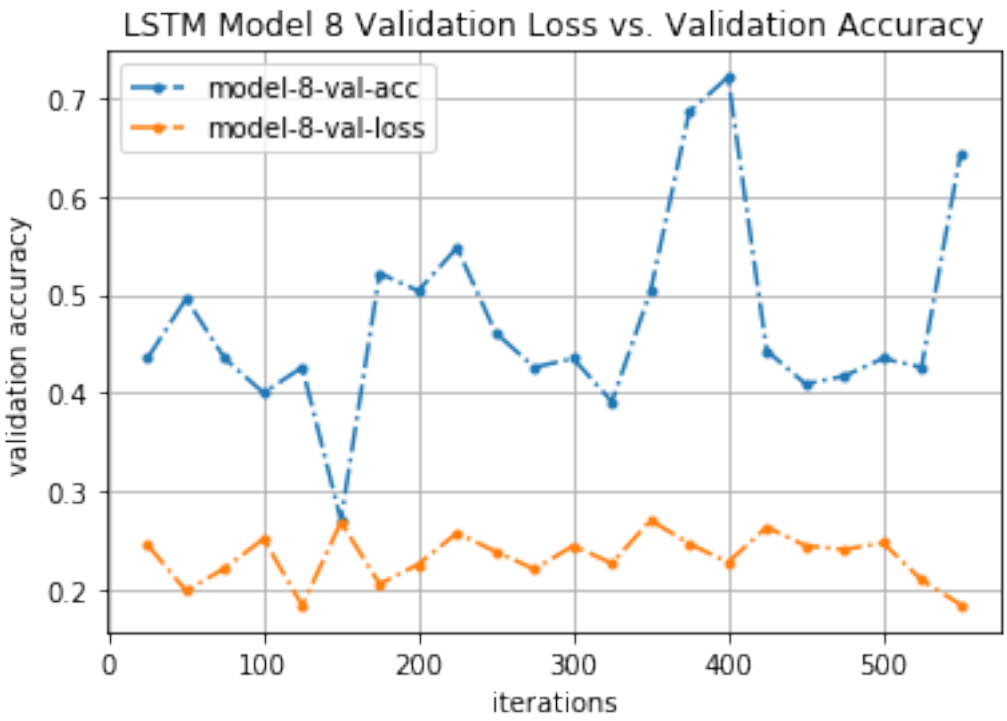
```
c:\python36\lib\site-packages\ipykernel_launcher.py:19: DeprecationWarning: `spline` is deprecated
spline is deprecated in scipy 0.19.0, use Bspline class instead.
```

```
c:\python36\lib\site-packages\scipy\interpolate\interpolate.py:2752: LinAlgWarning: scipy.linal
Ill-conditioned matrix detected. Result is not guaranteed to be accurate.
Reciprocal condition number5.244205e-17
  p = scipy.linalg.solve(Q, tmp)
```



In [ ]:

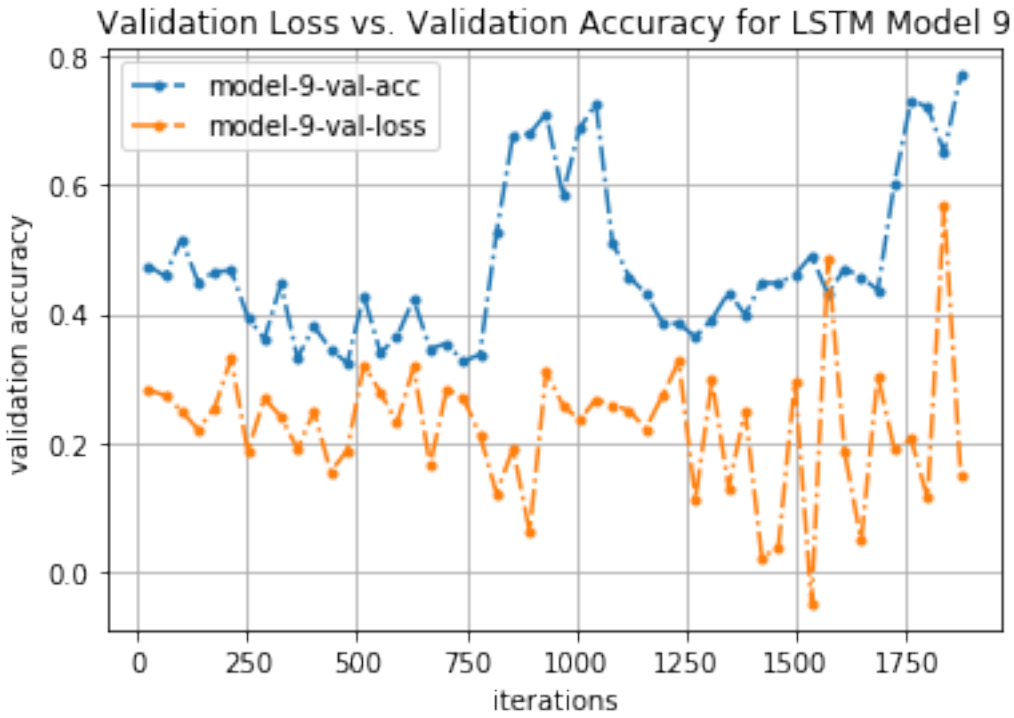
In [11]:
```
# lstm best model validation loss vs validation accuracy:

    #plt.figure(figsize=(30,15))
    plt.title('LSTM Model 8 Validation Loss vs. Validation Accuracy')
    plt.xlabel('iterations')
    plt.ylabel('validation accuracy')

    plt.plot(x_axis_8, best_model_val_acc_2, marker='o', markersize=3, linestyle=linestyle
    plt.plot(x_axis_8, best_model_val_loss_2, marker='o', markersize=3, linestyle=linesty

    plt.grid(True)
    plt.legend()
    plt.show()
```



In [12]:
```
smoothing_factor = 50
linestyle='-.'

#plt.figure(figsize=(30,15))
plt.title('Validation Loss vs. Validation Accuracy for LSTM Model 9')
plt.xlabel('iterations')
plt.ylabel('validation accuracy')
x = x_axis_9

y = best_model_val_acc_1
x_sm = np.array(x)
y_sm = np.array(y)
x_smooth = np.linspace(x_sm.min(), x_sm.max(), smoothing_factor)
y_smooth = spline(x, y, x_smooth)
plt.plot(x_smooth, y_smooth, marker='o', markersize=3, linestyle=linestyle, label='mod

y = best_model_val_loss_1
x_sm = np.array(x)
y_sm = np.array(y)
x_smooth = np.linspace(x_sm.min(), x_sm.max(), smoothing_factor)
y_smooth = spline(x, y, x_smooth)
plt.plot(x_smooth, y_smooth, marker='o', markersize=3, linestyle=linestyle, label='mod

plt.grid(True)
```

```
        plt.legend()
        plt.savefig('images\lstm_image_2_1_best_model_9_validation_loss_vs_validation _accura
        plt.show()
```

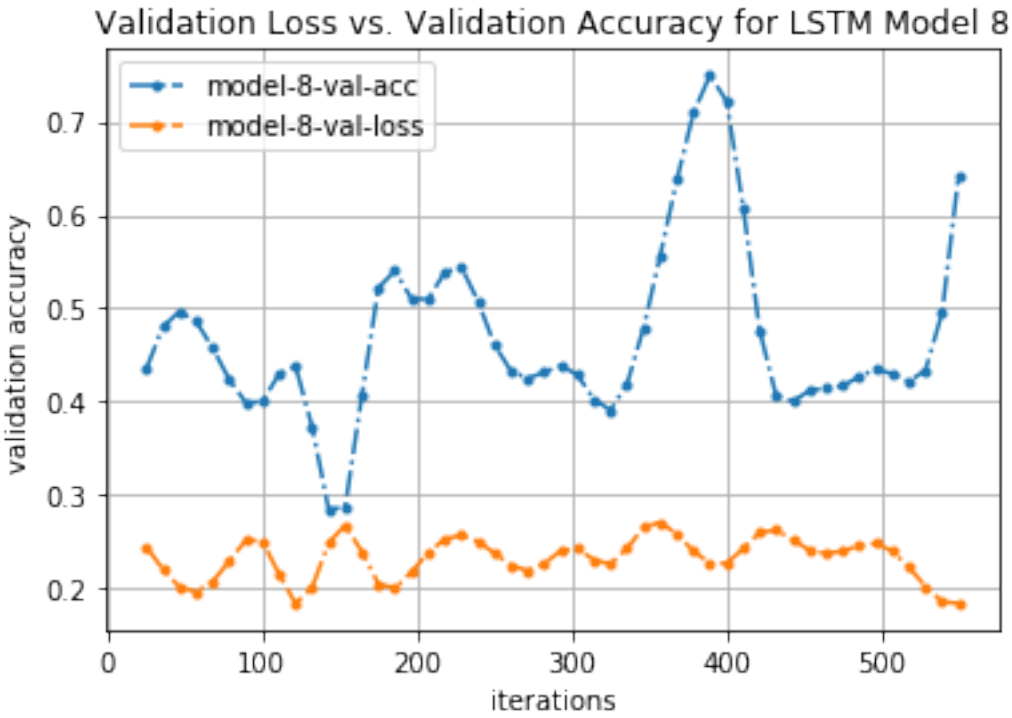
```
c:\python36\lib\site-packages\ipykernel_launcher.py:14: DeprecationWarning: `spline` is deprec
spline is deprecated in scipy 0.19.0, use Bspline class instead.

c:\python36\lib\site-packages\ipykernel_launcher.py:21: DeprecationWarning: `spline` is deprec
spline is deprecated in scipy 0.19.0, use Bspline class instead.
```



In [13]:
```
        smoothing_factor = 50
        linestyle='-.'

        #plt.figure(figsize=(30,15))
        plt.title('Validation Loss vs. Validation Accuracy for LSTM Model 8')
        plt.xlabel('iterations')
        plt.ylabel('validation accuracy')
        x = x_axis_8

        y = best_model_val_acc_2
        x_sm = np.array(x)
        y_sm = np.array(y)
        x_smooth = np.linspace(x_sm.min(), x_sm.max(), smoothing_factor)
```

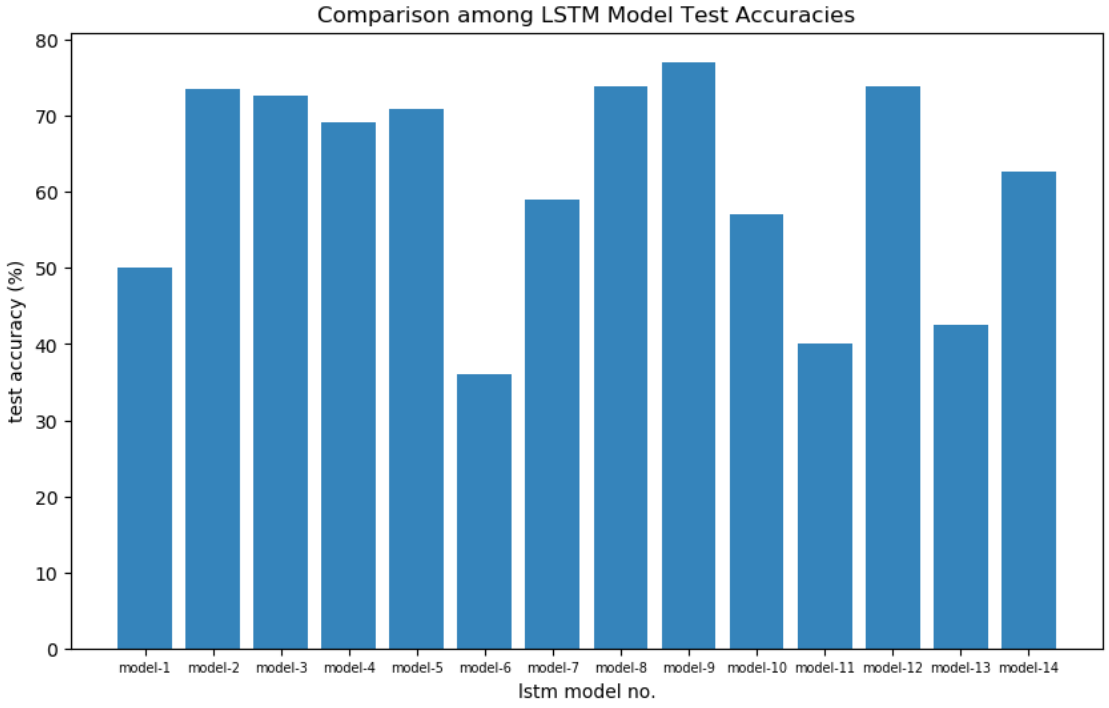
```
        y_smooth = spline(x, y, x_smooth)
        plt.plot(x_smooth, y_smooth, marker='o', markersize=3, linestyle=linestyle, label='mo

        y = best_model_val_loss_2
        x_sm = np.array(x)
        y_sm = np.array(y)
        x_smooth = np.linspace(x_sm.min(), x_sm.max(), smoothing_factor)
        y_smooth = spline(x, y, x_smooth)
        plt.plot(x_smooth, y_smooth, marker='o', markersize=3, linestyle=linestyle, label='mo

        plt.grid(True)
        plt.legend()
        plt.savefig('images\lstm_image_2_2_best_model_8_validation_loss_vs_validation _accura
        plt.show()
```

```
c:\python36\lib\site-packages\ipykernel_launcher.py:14: DeprecationWarning: `spline` is deprec
spline is deprecated in scipy 0.19.0, use Bspline class instead.

c:\python36\lib\site-packages\ipykernel_launcher.py:21: DeprecationWarning: `spline` is deprec
spline is deprecated in scipy 0.19.0, use Bspline class instead.
```



In [ ]:

In [14]:
```
# comparing lstm model test accuracies (bar chart):

plt.rcdefaults()

plt.figure(figsize=(10,6))
objects = []
for i in range(0,len(model_list)):
    object_name = "model-"+str(i+1)
    objects.append(object_name)
y_pos = np.arange(len(objects))
performance = test_acc_list ###

plt.bar(y_pos, performance, align='center', alpha=0.9)
plt.xticks(y_pos, objects)
#plt.tick_params(axis='both', which='major', labelsize=10)
plt.tick_params(axis='x', which='major', labelsize=7)
plt.xlabel('lstm model no.')
plt.ylabel('test accuracy (%)')
plt.title('Comparison among LSTM Model Test Accuracies')
plt.savefig('images\lstm_image_3_comparing_lstm_model_test_accuracies_bar_chart.png',
plt.show()
```



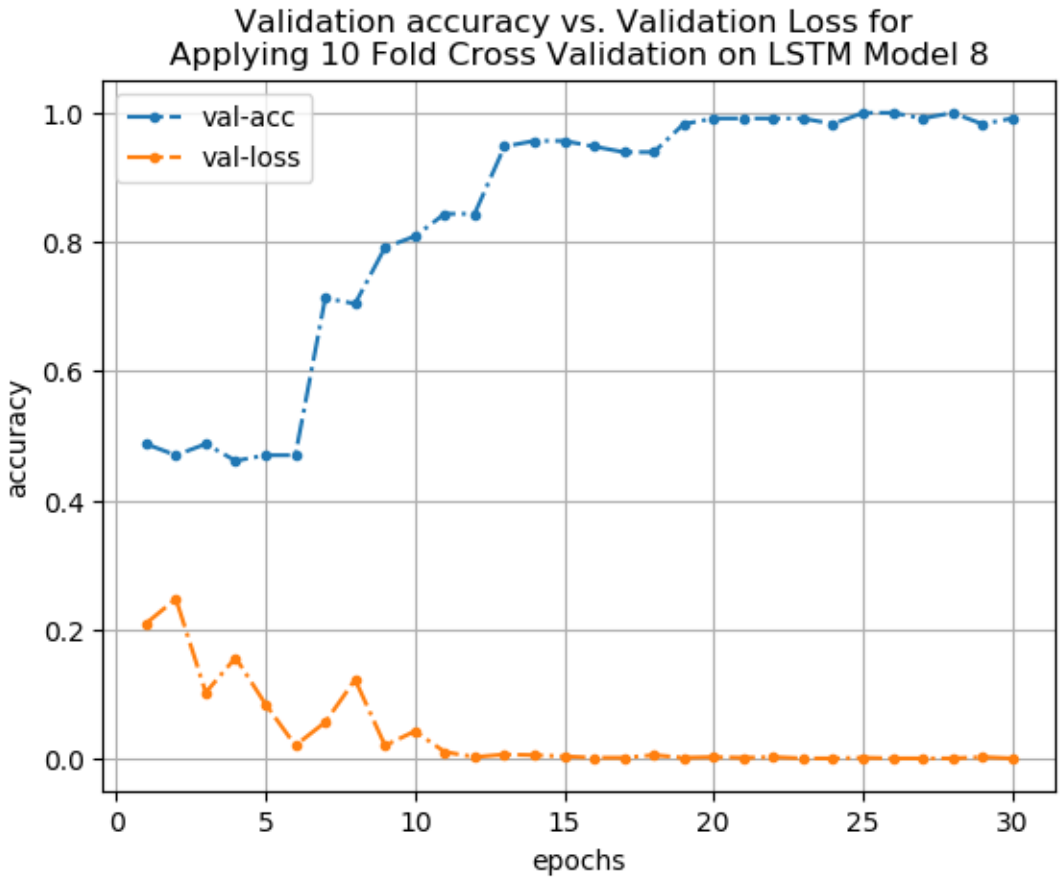
In [ ]:

In [15]:
```
# lstm_10_fold_cross_validation_on_model_8:
lstm_10_fold_cross_val_acc_list = [0.4870,0.4696,0.4870,0.4609,0.4696,0.4696,0.7130,0
```

```
lstm_10_fold_cross_val_loss_list = [0.209,0.247,0.102,0.156,0.084,0.020,0.056,0.122,0

print(len(lstm_10_fold_cross_val_acc_list))
print(len(lstm_10_fold_cross_val_loss_list))
```

```
30
30
```

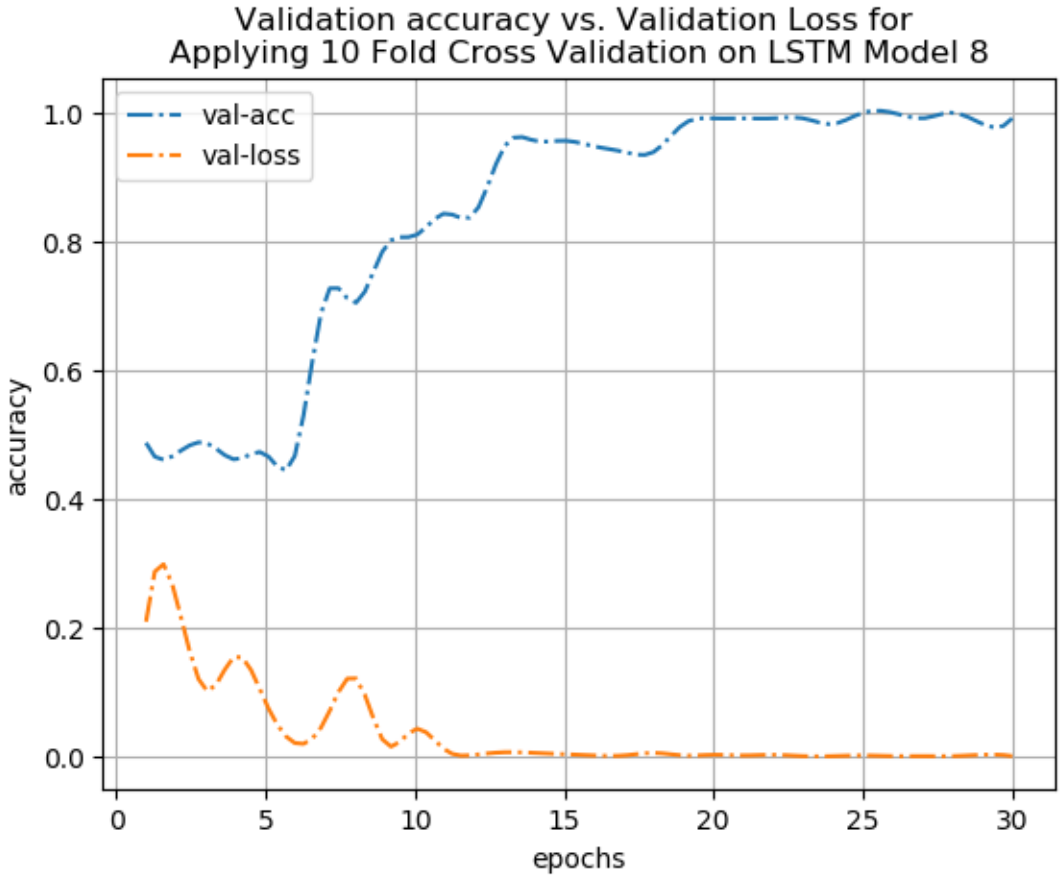
In [16]:
```
smoothing_factor = 200
linestyle='-.'

#plt.figure(figsize=(30,15))
plt.title('Validation accuracy vs. Validation Loss for \nApplying 10 Fold Cross Valida
plt.xlabel('epochs')
plt.ylabel('accuracy')

x = [i for i in range (1,len(lstm_10_fold_cross_val_acc_list)+1)]
y = lstm_10_fold_cross_val_acc_list
plt.plot(x, y, marker='o', markersize=3, linestyle=linestyle, label='val-acc')

y = lstm_10_fold_cross_val_loss_list
plt.plot(x, y, marker='o', markersize=3, linestyle=linestyle, label='val-loss')

plt.grid(True)
plt.legend()
#plt.savefig('images\lstm_image_4_accuracy_vs_loss_for_10_fold_cross_validation.png',
plt.show()
```



In [17]:
```
smoothing_factor = 100
linestyle='-.'

#plt.figure(figsize=(30,15))
plt.title('Validation accuracy vs. Validation Loss for \nApplying 10 Fold Cross Valida
plt.xlabel('epochs')
plt.ylabel('accuracy')
x = [i for i in range (1,len(lstm_10_fold_cross_val_acc_list)+1)]

y = lstm_10_fold_cross_val_acc_list
x_sm = np.array(x)
y_sm = np.array(y)
x_smooth = np.linspace(x_sm.min(), x_sm.max(), smoothing_factor)
y_smooth = spline(x, y, x_smooth)
plt.plot(x_smooth, y_smooth, marker='', markersize=1, linestyle=linestyle, label='val-

y = lstm_10_fold_cross_val_loss_list
x_sm = np.array(x)
y_sm = np.array(y)
```

```
        x_smooth = np.linspace(x_sm.min(), x_sm.max(), smoothing_factor)
        y_smooth = spline(x, y, x_smooth)
        plt.plot(x_smooth, y_smooth, marker='', markersize=1, linestyle=linestyle, label='val-

        plt.grid(True)
        plt.legend()
        plt.savefig('images\lstm_image_4_accuracy_vs_loss_for_10_fold_cross_validation.png', b
        plt.show()
```

```
c:\python36\lib\site-packages\ipykernel_launcher.py:14: DeprecationWarning: `spline` is deprecat
spline is deprecated in scipy 0.19.0, use Bspline class instead.

c:\python36\lib\site-packages\ipykernel_launcher.py:21: DeprecationWarning: `spline` is deprecat
spline is deprecated in scipy 0.19.0, use Bspline class instead.
```



In [18]:
```
# LSTM 10 FOLD CROSS VALIDATION MODEL ACCURACY::
lstm_10_fold_cross_val_folds_acc_list = [0.4870,0.4696,0.7913,0.8435,0.9565,0.9391,0.9
lstm_10_fold_cross_val_model_acc = sum(lstm_10_fold_cross_val_folds_acc_list)/len(lstm
print('LSTM 10 FOLD CROSS VALIDATION MODEL ACCURACY =',lstm_10_fold_cross_val_model_ac
```

```
LSTM 10 FOLD CROSS VALIDATION MODEL ACCURACY = 0.84435
```

In [ ]: