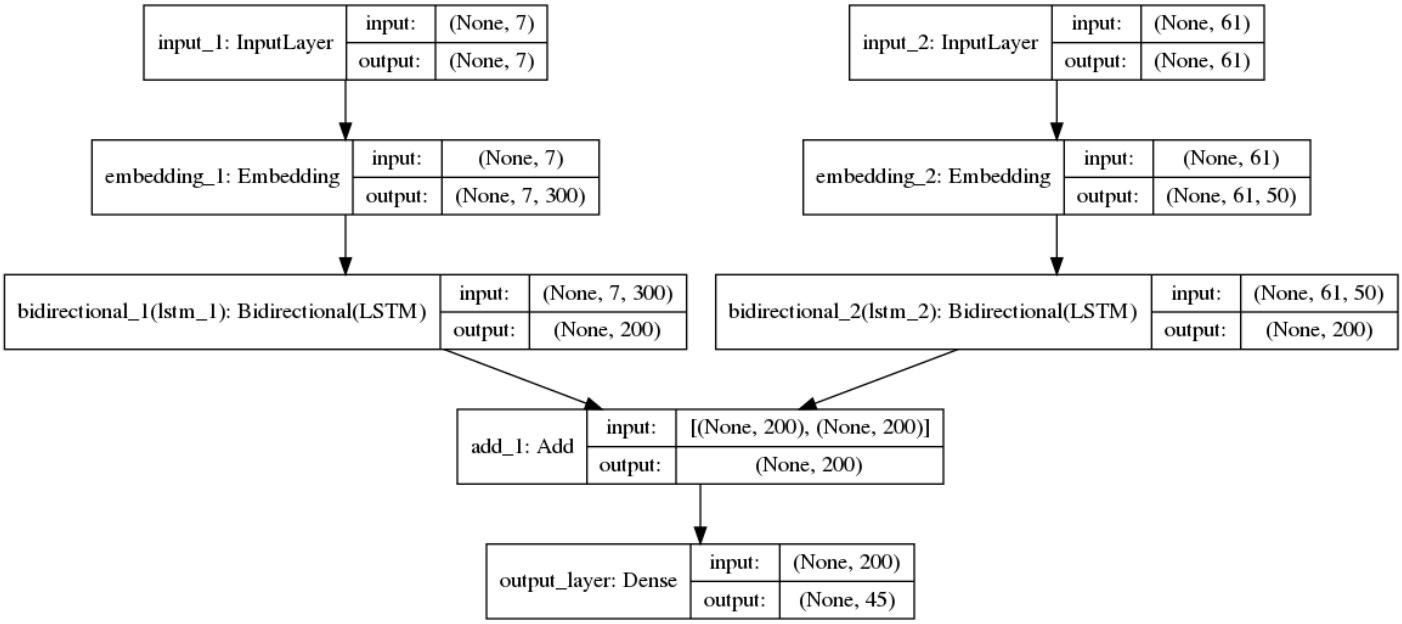
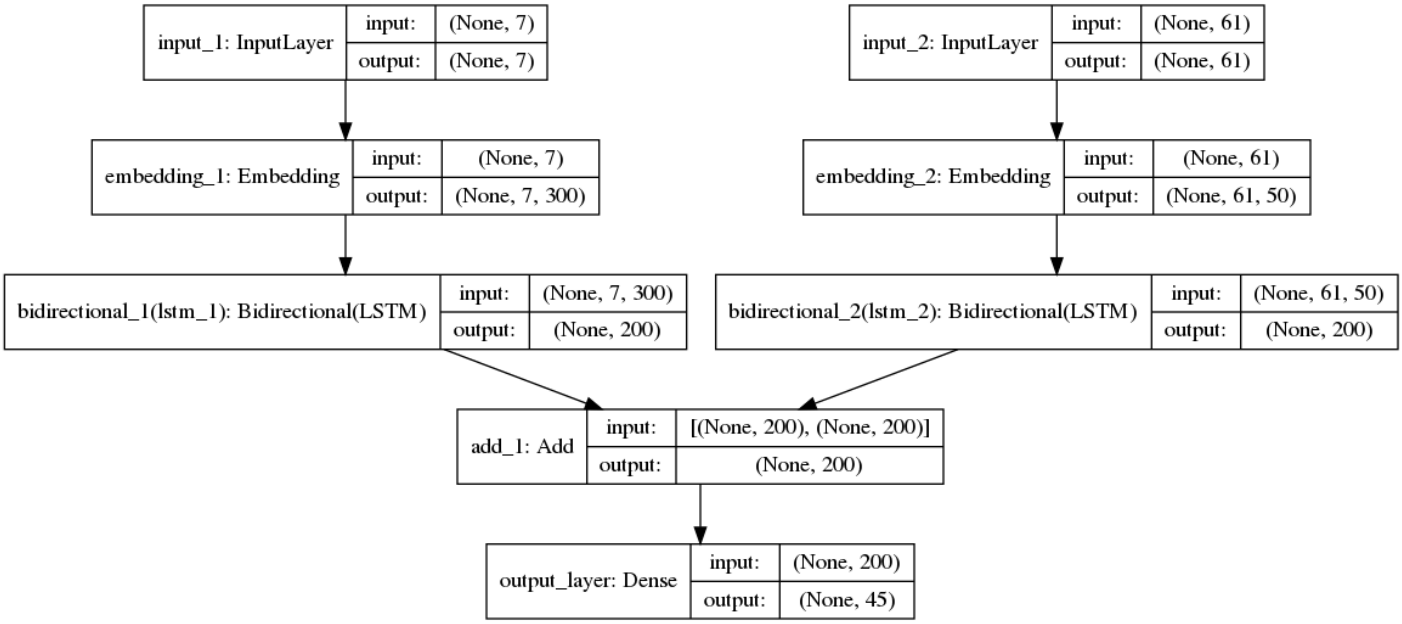
# Define Models in Keras: without functional API and with functional API



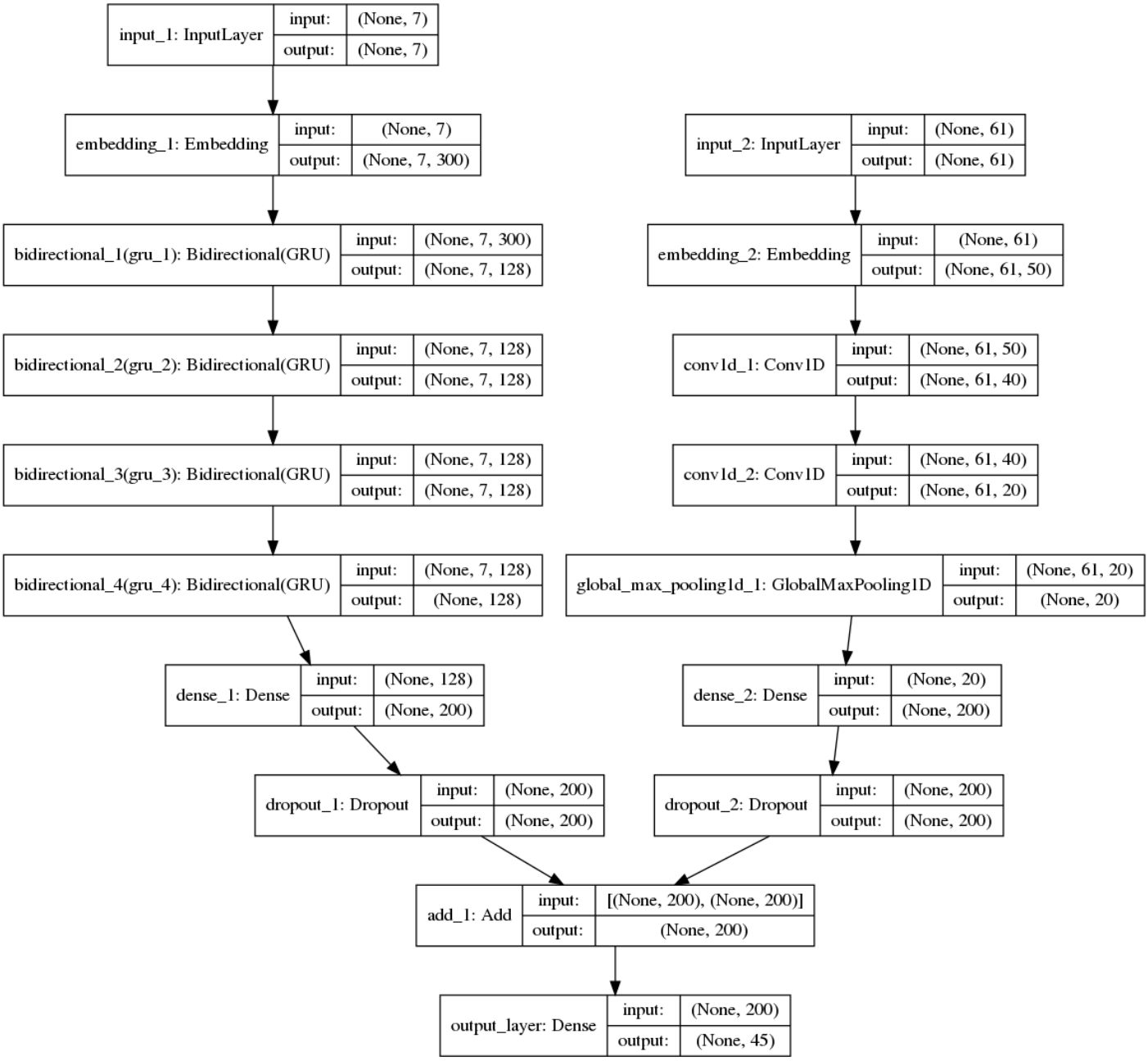

## Keras Example, not in Keras functional API

```
## Word model

word_model=Sequential()
word_model.add(Embedding(input_dim=n_words, output_dim=300, input_length=context_window,
                         mask_zero=True))
word_model.add(Bidirectional(GRU(100, return_sequences=False)))
word_model.summary()

## Character_model

character_model=Sequential()
character_model.add(Embedding(n_chars, 50, input_length=MAX_LEN, mask_zero=True,
                    embeddings_initializer=RandomUniform(seed=1234)))
character_model.add(Bidirectional(LSTM(100,return_sequences=False)))
character_model.summary()

## Merge two models

merge_word_character_pos_tag_model = Sequential()
merge_word_character_pos_tag_model.add(Merge([character_model, word_model]))
merge_word_character_pos_tag_model.add(Dense(n_tags))
merge_word_character_pos_tag_model.add(Activation('softmax'))
merge_word_character_pos_tag_model.summary()
```

## The same Keras Example, Keras functional API




```
## Word model

word_model_in = Input(shape=(context_window,))
word_model_out = Embedding(input_dim=n_words, output_dim=300, input_length=context_window,
                       mask_zero=True)(word_model_in)
word_model_out1 = Bidirectional(LSTM(100,input_shape=(None,3,300),
                       return_sequences=False))(word_model_out)
word_model = Model(word_model_in, word_model_out1)

## Character_model

char_model_in = Input(shape=(MAX_LEN,))
char_model_out = Embedding(n_chars, 50, input_length=MAX_LEN, mask_zero=True,
                   embeddings_initializer=RandomUniform(seed=1234))(char_model_in)
char_model_out1 = Bidirectional(LSTM(100, input_shape=(None,61,50),
                   return_sequences=False))(char_model_out)
char_model = Model(char_model_in, char_model_out1)

## Merge two models

merged_model = add([word_model_out1,char_model_out1])
out = Dense(n_tags, activation='softmax', name='output_layer')(merged_model)
merge_word_character_model = Model([word_model_in, char_model_in], out)
```


input_1: InputLayer
input: (None, 7)
output: (None, 7)
embedding_1: Embedding
input: (None, 7)
output: (None, 7, 300)
bidirectional_1(gru_1): Bidirectional(GRU)
input: (None, 7, 300)
output: (None, 7, 128)
bidirectional_2(gru_2): Bidirectional(GRU)
input: (None, 7, 128)
output: (None, 7, 128)
bidirectional_3(gru_3): Bidirectional(GRU)
input: (None, 7, 128)
output: (None, 7, 128)
bidirectional_4(gru_4): Bidirectional(GRU)
input: (None, 7, 128)
output: (None, 128)
dense_1: Dense
input: (None, 128)
output: (None, 200)
dropout_1: Dropout
input: (None, 200)
output: (None, 200)
input_2: InputLayer
input: (None, 61)
output: (None, 61)
embedding_2: Embedding
input: (None, 61)
output: (None, 61, 50)
conv1d_1: Conv1D
input: (None, 61, 50)
output: (None, 61, 40)
conv1d_2: Conv1D
input: (None, 61, 40)
output: (None, 61, 20)
global_max_pooling1d_1: GlobalMaxPooling1D
input: (None, 61, 20)
output: (None, 20)
dense_2: Dense
input: (None, 20)
output: (None, 200)
dropout_2: Dropout
input: (None, 200)
output: (None, 200)
add_1: Add
input: [(None, 200), (None, 200)]
output: (None, 200)
output_layer: Dense
input: (None, 200)
output: (None, 45)

```
In [ ]: ## Word model

word_model_in = Input(shape=(context_window,))
word_model =
        Embedding(output_dim=300,
        input_dim=n_words,
        input_length=context_window,
        mask_zero=True)(word_model_in)
word_model = Bidirectional(GRU(64, return_sequences=True), merge_mode='concat')(word_model)
word_model = Bidirectional(GRU(64, return_sequences=True), merge_mode='concat')(word_model)
word_model = Bidirectional(GRU(64, return_sequences=True), merge_mode='concat')(word_model)
word_model = Bidirectional(GRU(64, return_sequences=False), merge_mode='concat')(word_model)
word_model = Dense(200, init='uniform', activation='relu')(word_model)
word_model_out = Dropout(0.5)(word_model)
word_model = Model(word_model_in, word_model_out)
word_model.summary()

## Character model

character_model_in = Input(shape=(MAX_LEN,))
character_model =
          Embedding(n_chars, 50, input_length=MAX_LEN, mask_zero=False,
          embeddings_initializer=RandomUniform(seed=1234))(character_model_in)

character_model = Convolution1D(filters=40, kernel_size=4, padding='same')(character_model)
character_model = Convolution1D(filters=20, kernel_size=2, padding='same')(character_model)
character_model = GlobalMaxPooling1D()(character_model)

character_model = Dense(200, init='uniform', activation='relu')(character_model)
character_model_out = Dropout(0.5)(character_model)
character_model = Model(character_model_in, character_model_out)
character_model.summary()

## Merge two models


merged_model = add([word_model_out,character_model_out])

out = Dense(n_tags, activation='softmax', name='output_layer')(merged_model)

merge_word_character_model = Model([word_model_in, character_model_in], out)
```

# Homework: Build Keras Model according to the Model Visualization

(Due: 20 November, 10:30)

Take the data and the notebook we used in class (see the .ipynb and the .py files in this folder) and define the neural network model according to the scheme below.
I marked three cells where you should insert your code with " <your code here> "