

## [analysis]\_\_percent\_improvement\_examples

June 27, 2022

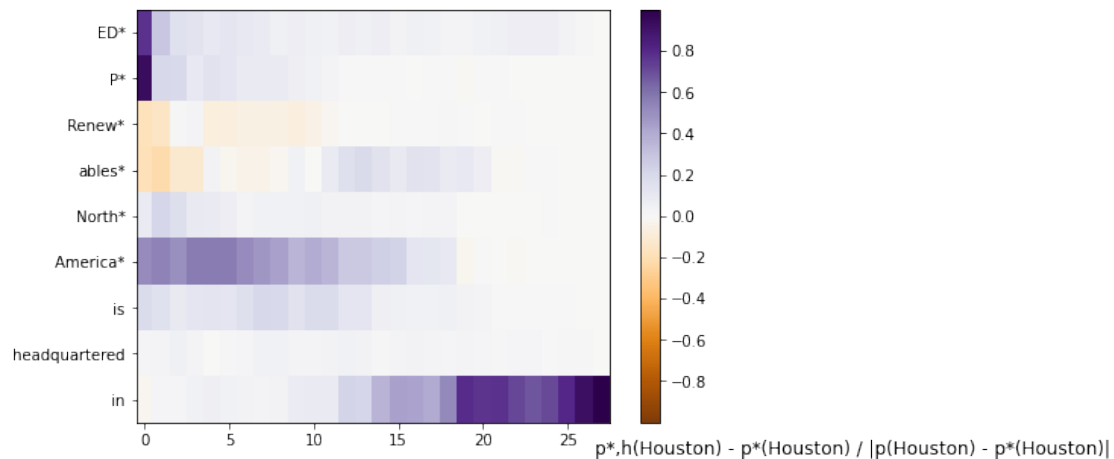
```
[6]: import torch
import random
```

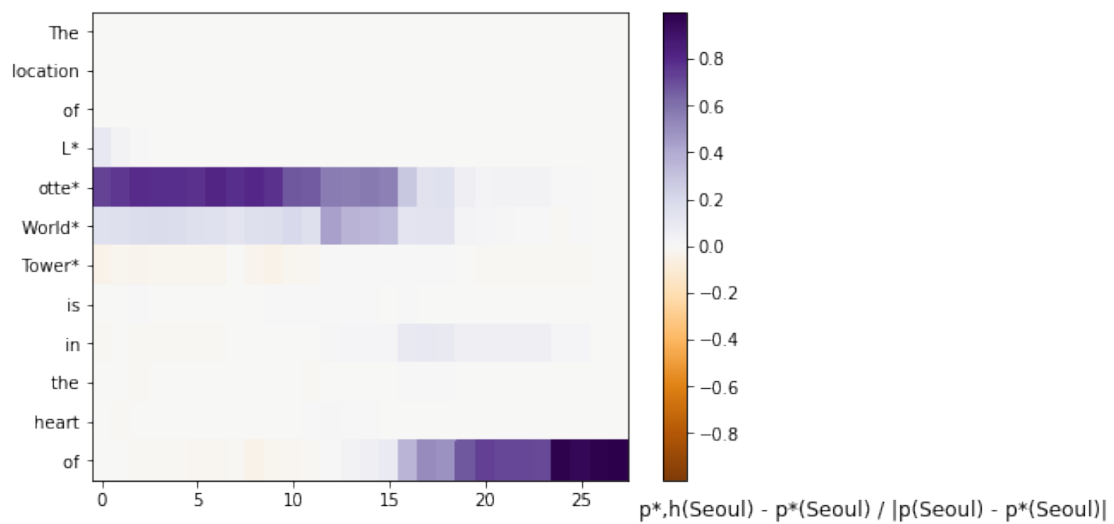
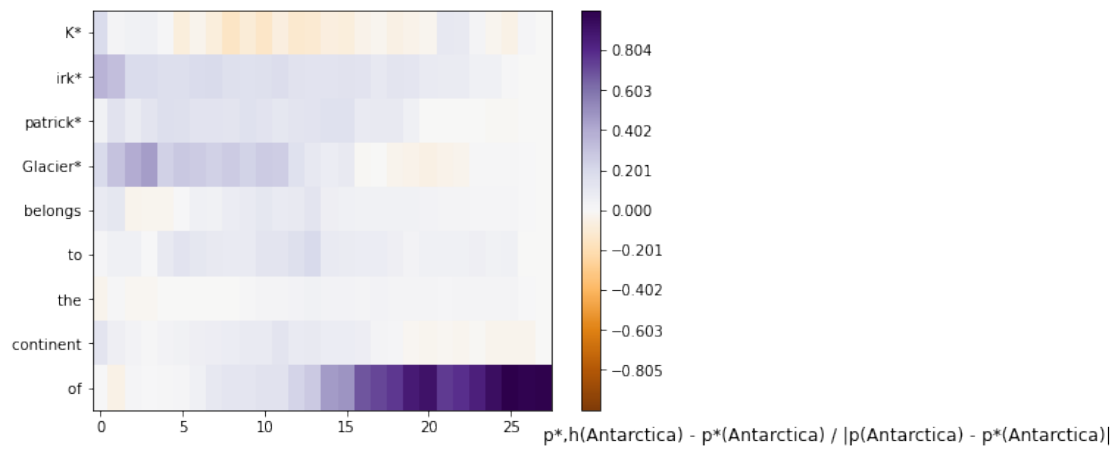
```
import sys
sys.path.append("../../lib")
import utility
```

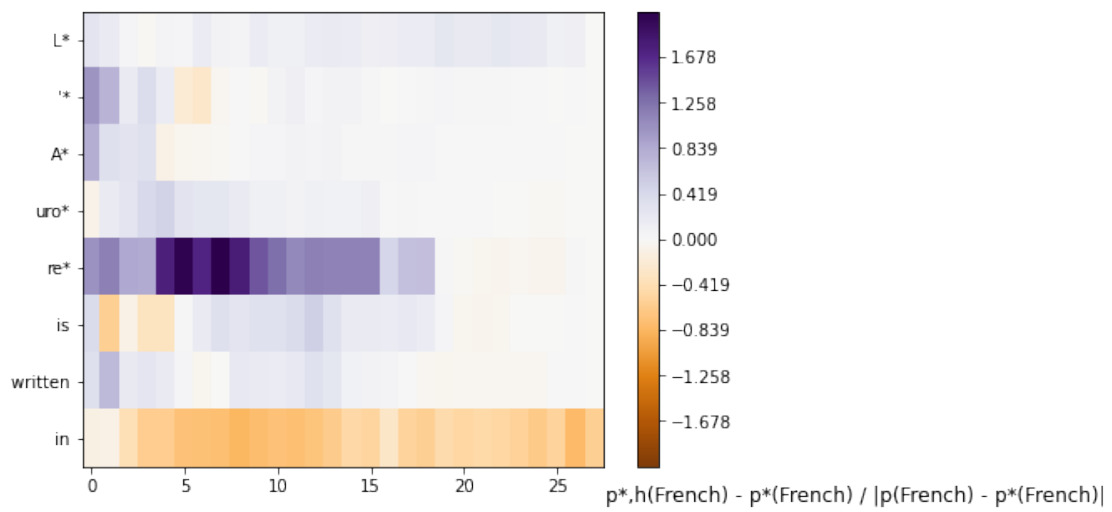
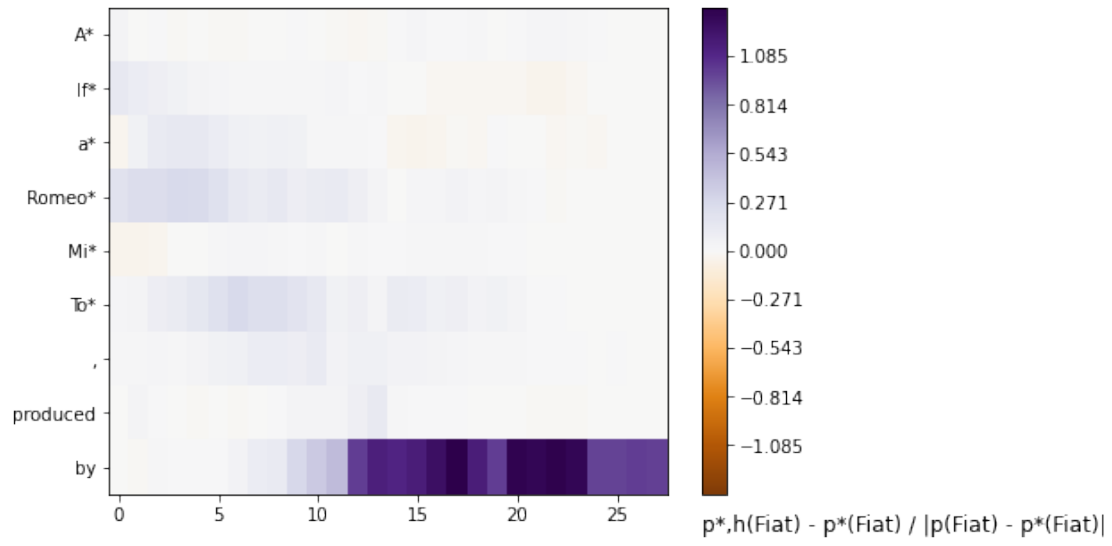
```
[7]: res = torch.load("../data/gpt-j/percent_improvement_1000_examples.pt")
res = random.sample(res, 100)
```

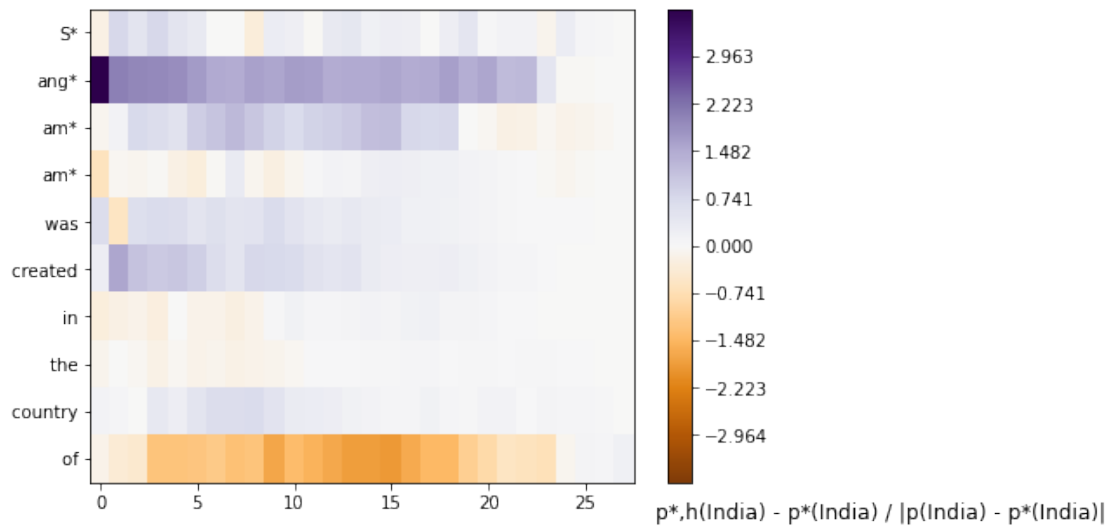
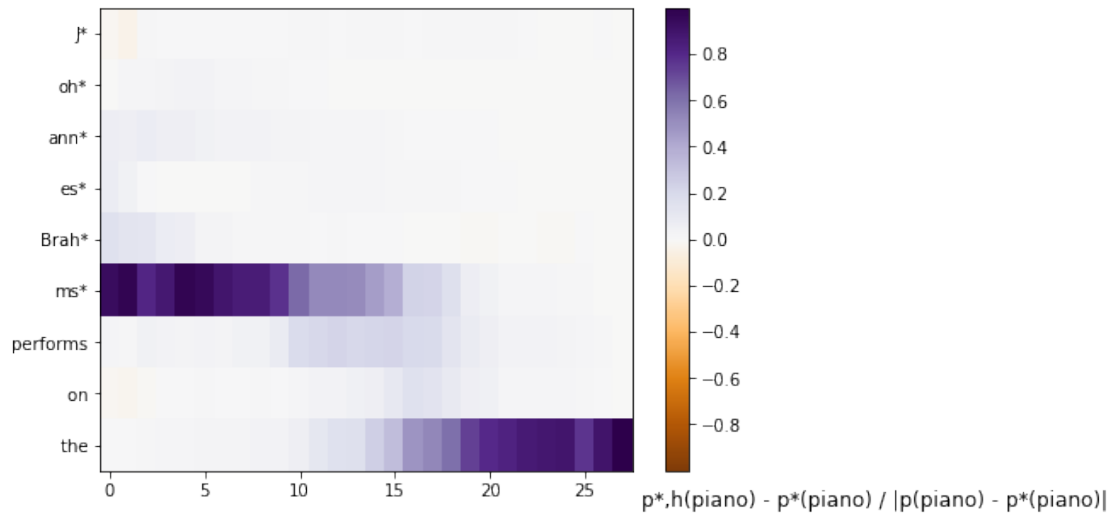
```
[8]: # no cutoff applied
for prob, access in res:
    text, tkens, num_layers, _, _ = access
    x = num_layers
    y = tkens
    c = 0.001
    bound = max(torch.max(prob).item(), abs(torch.min(prob).item()), c)
    incr = bound / 10000
    token = text.split()[-1]
    title = f"p*,h({token}) - p*({token}) / |p({token}) - p*({token})|"
    color_schema = "PuOr"

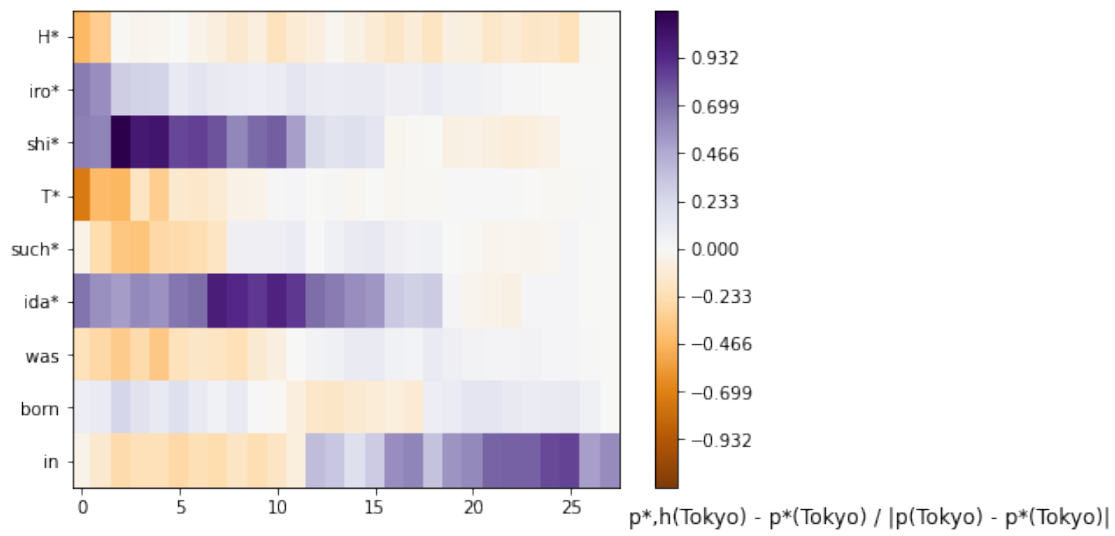
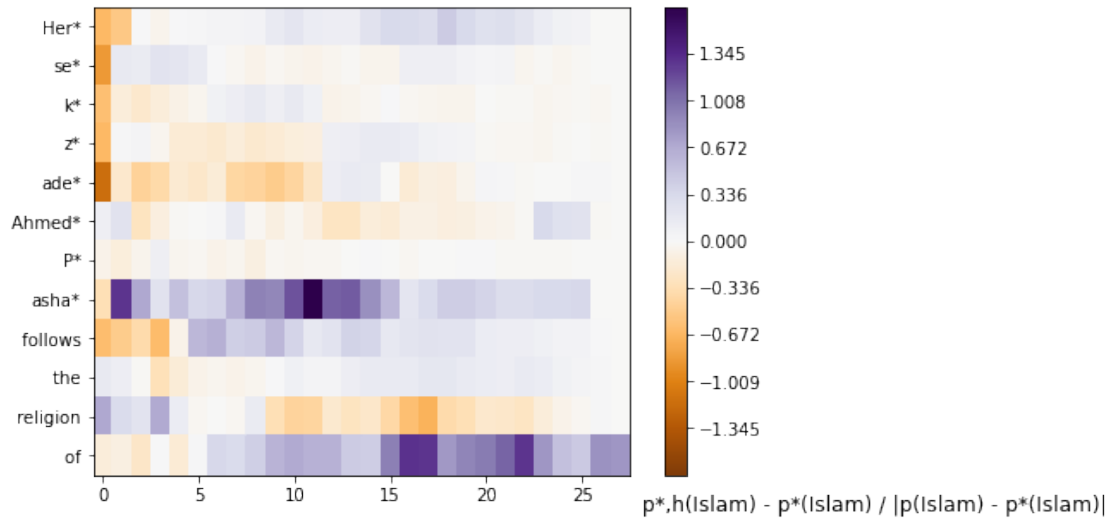
    utility.plot_results(prob, x, y, -bound, bound, incr, title, color_schema)
```

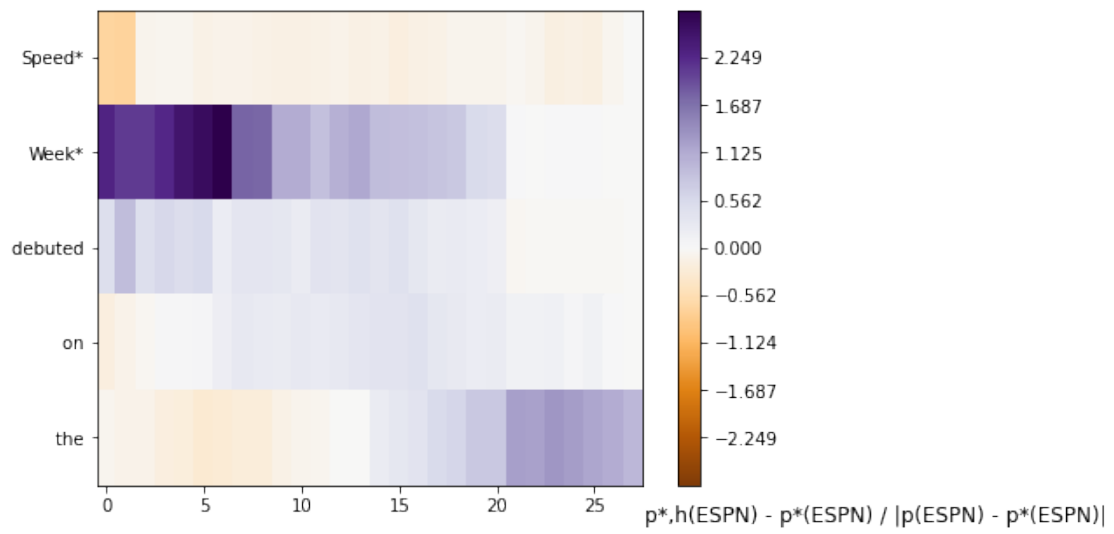
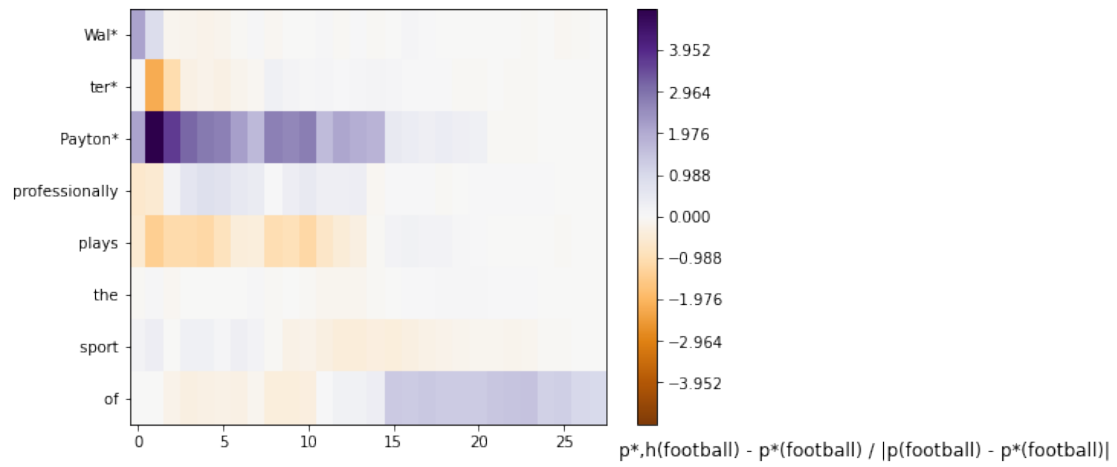


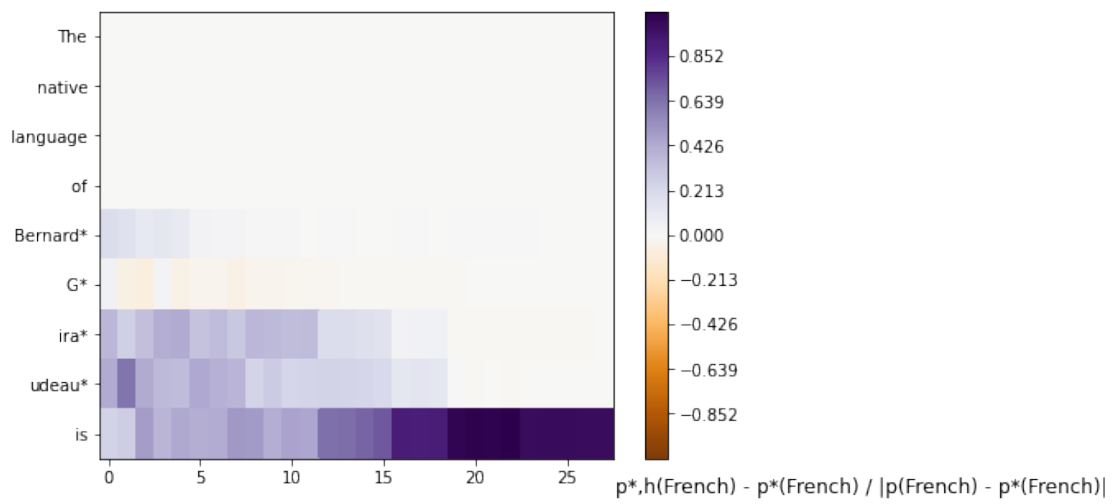
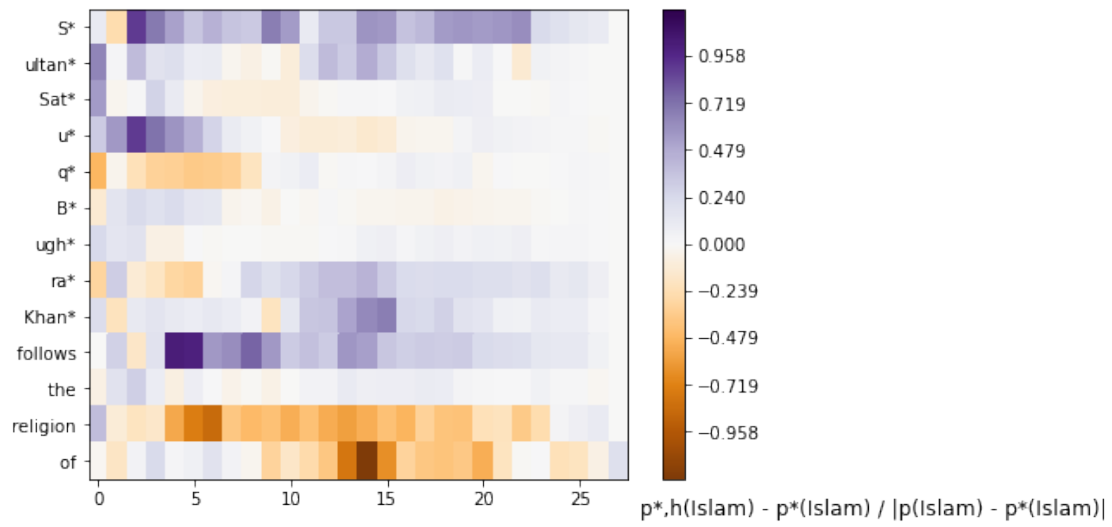


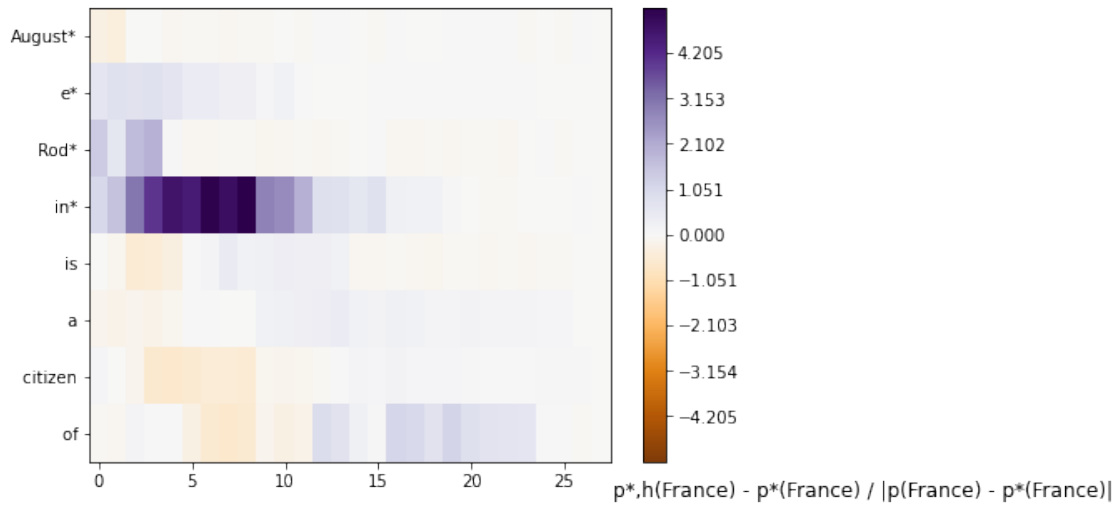
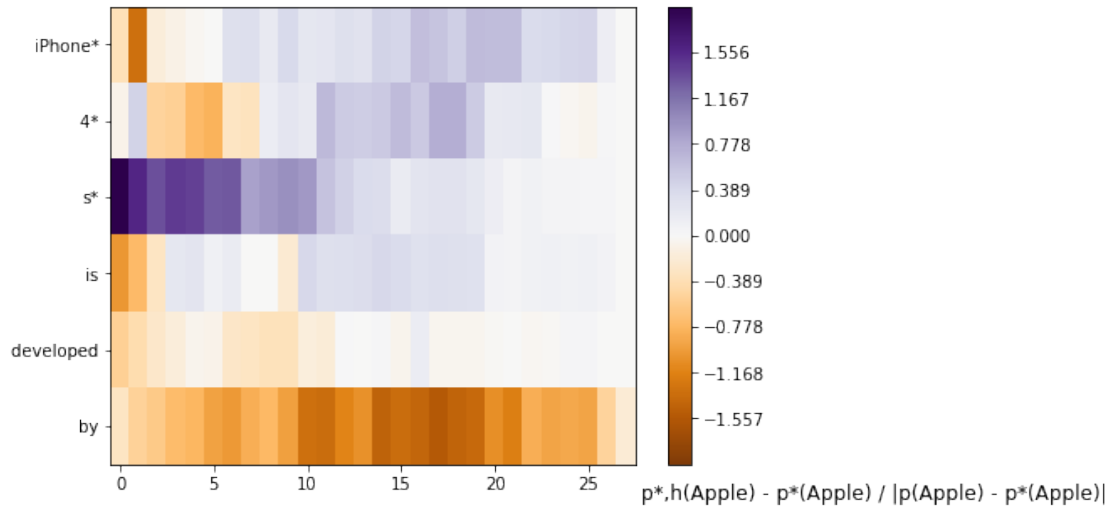




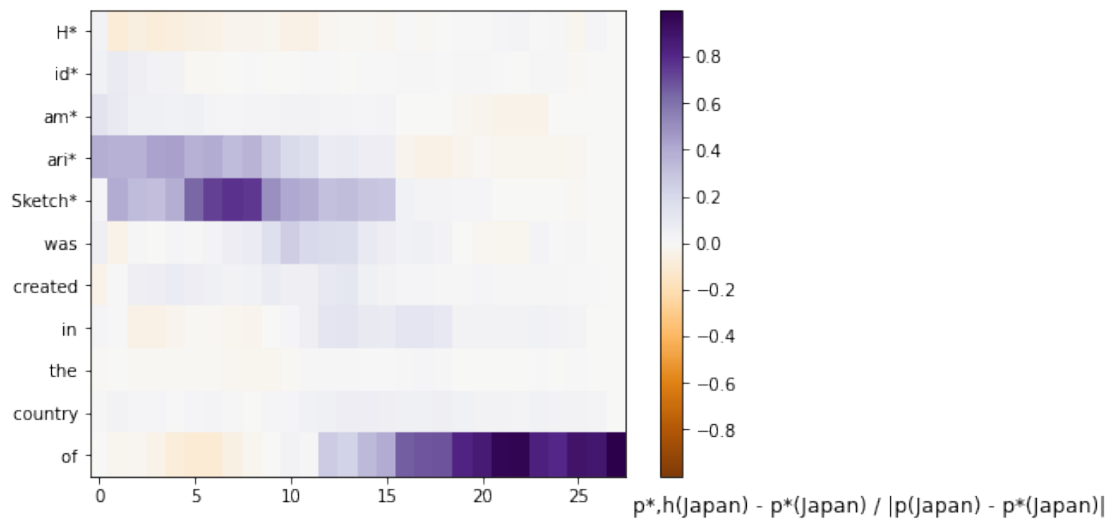
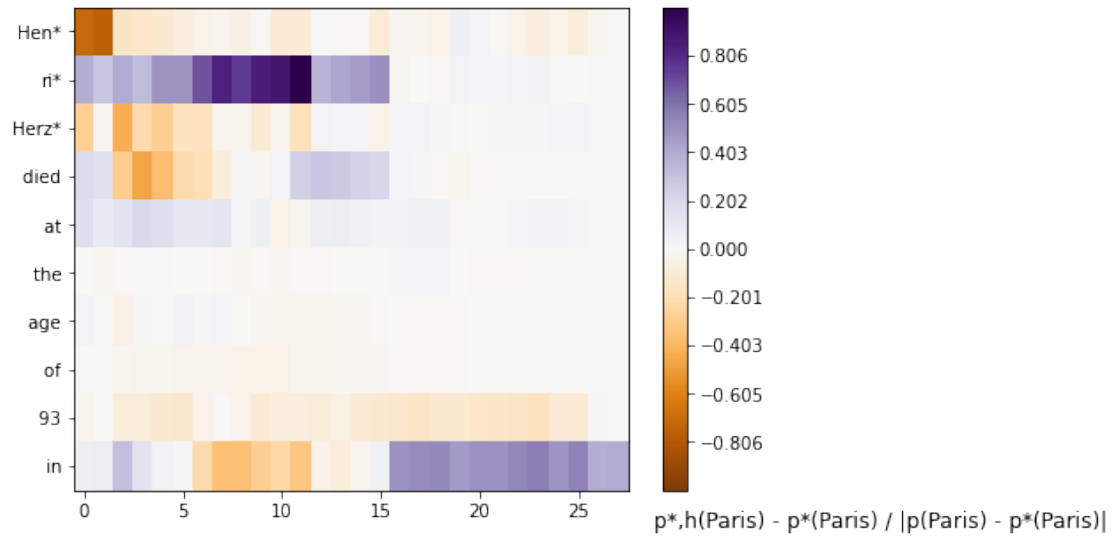


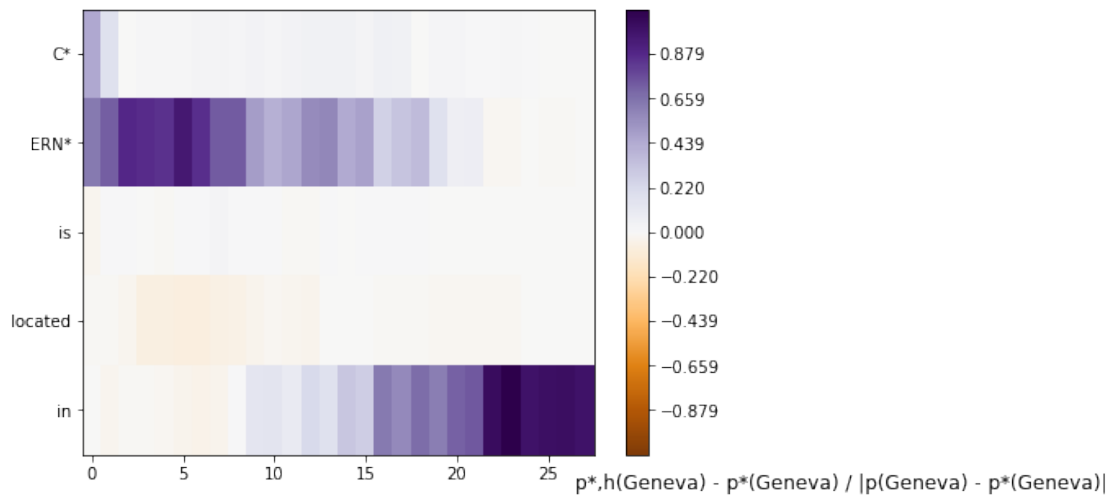
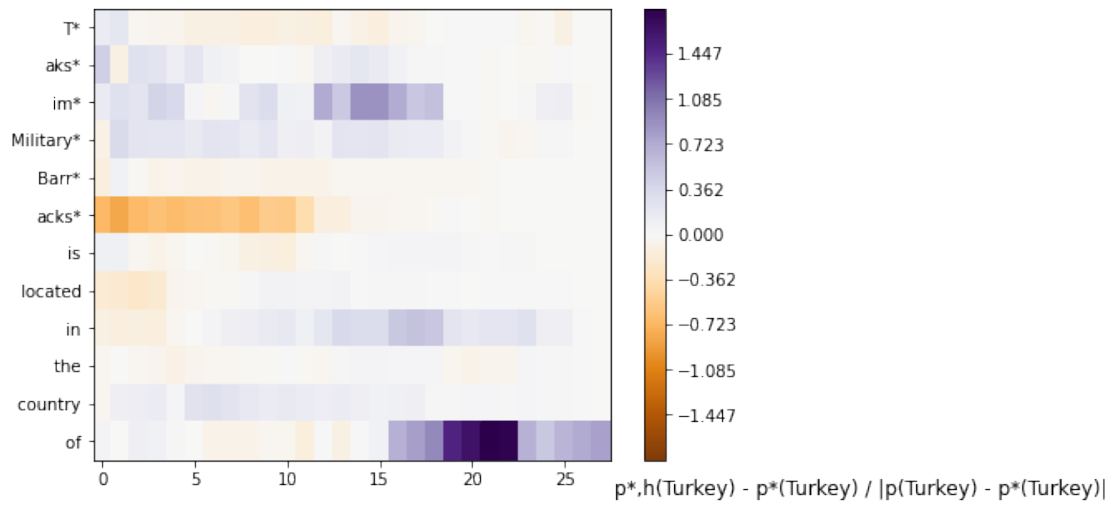


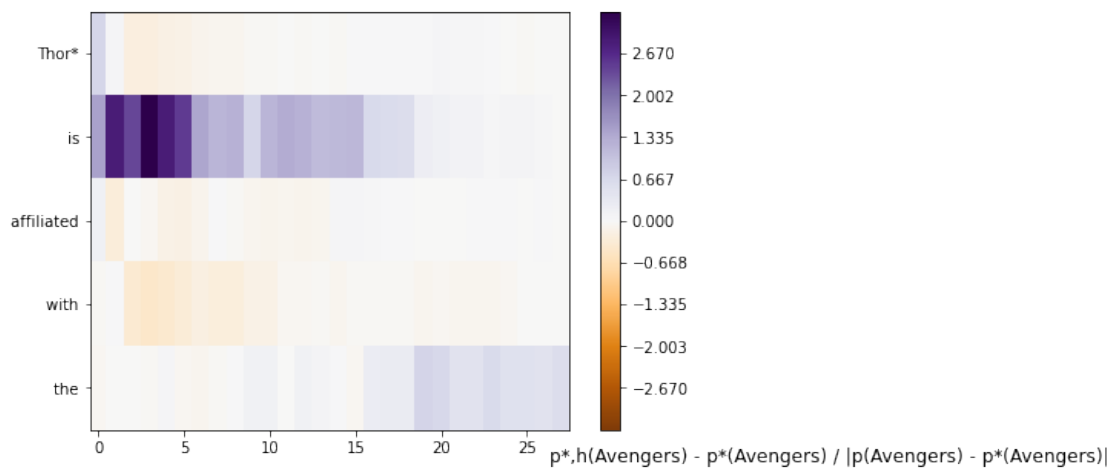
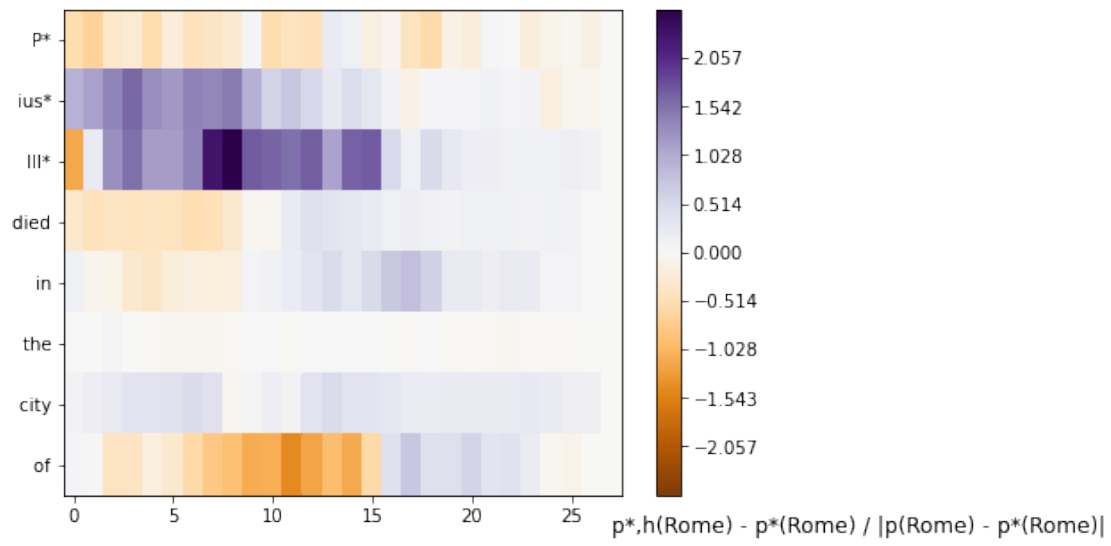


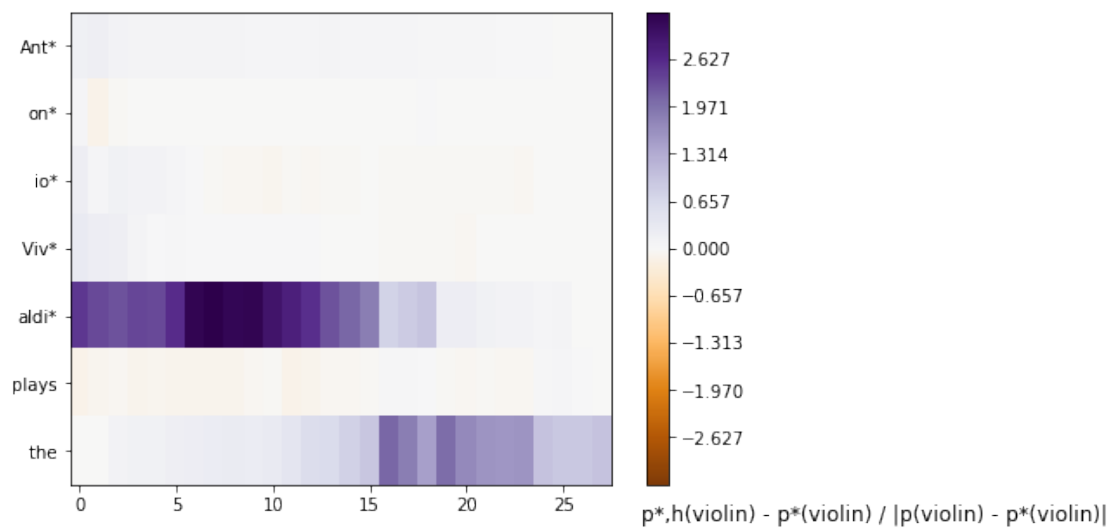
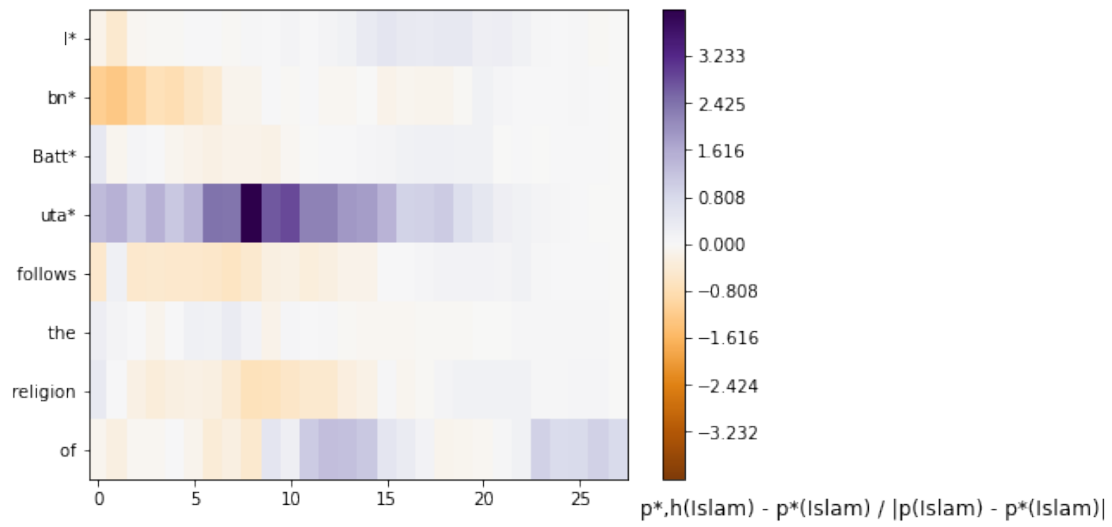


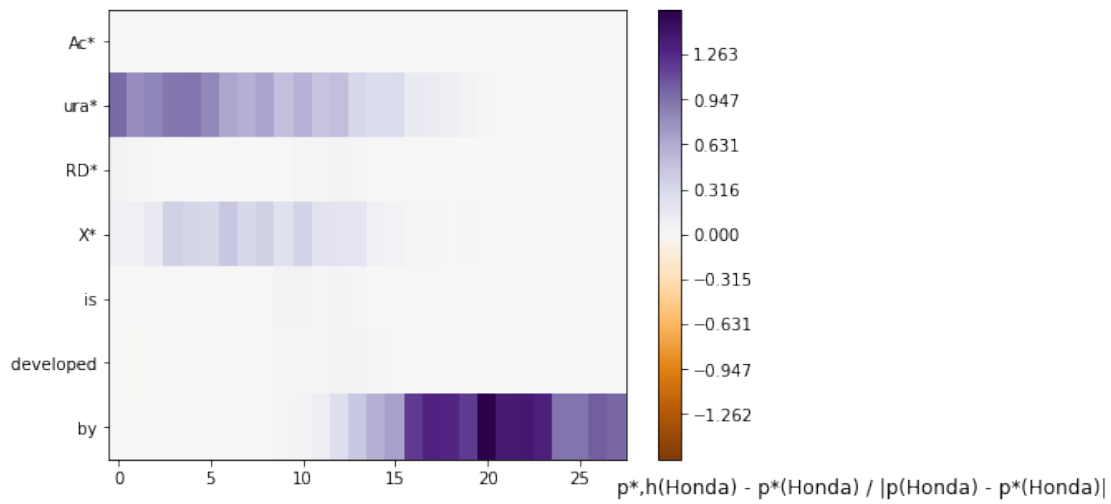
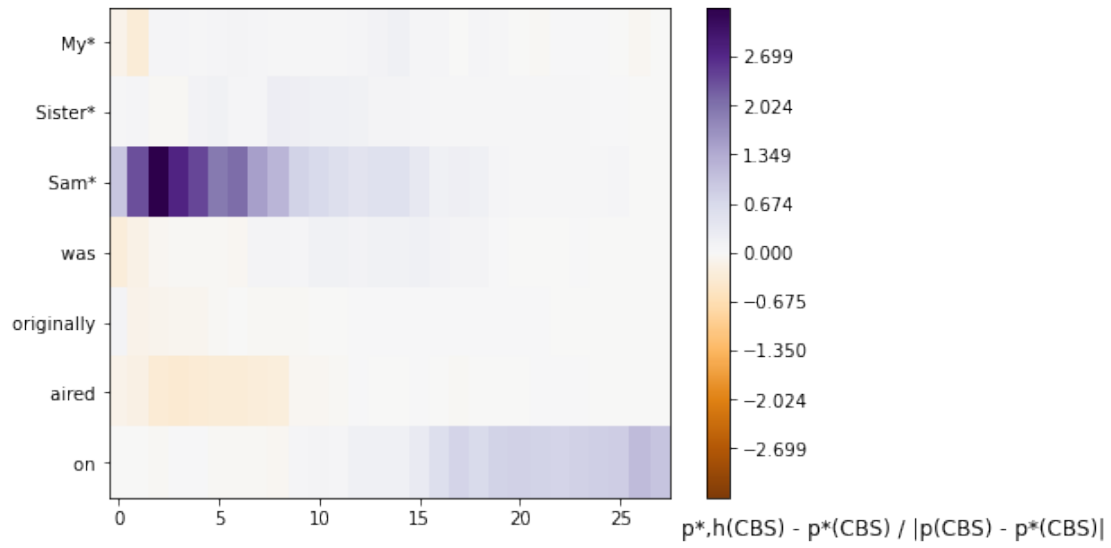


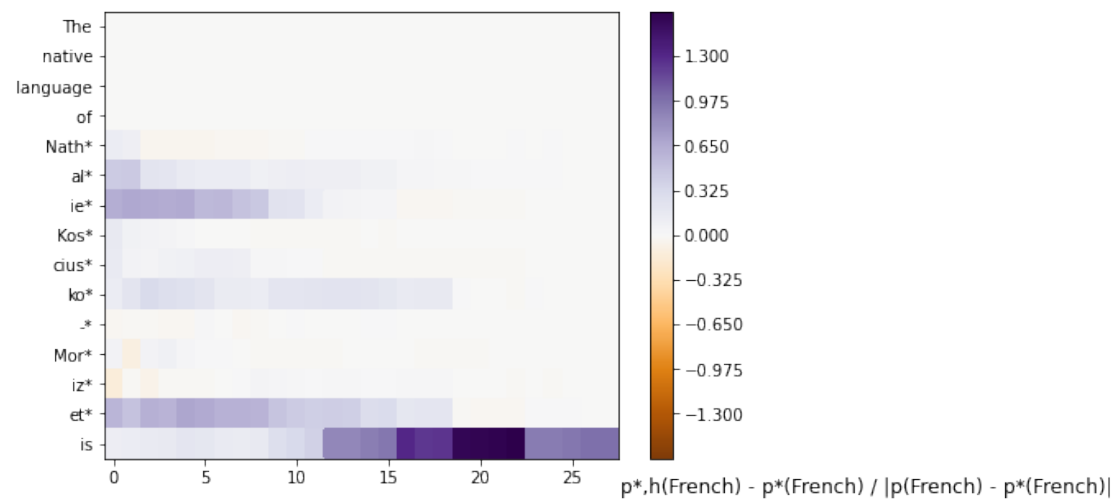
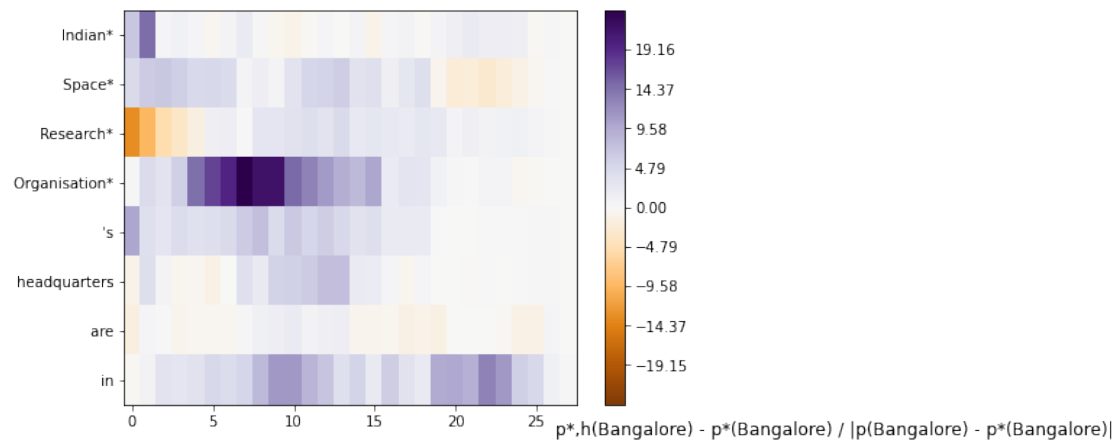
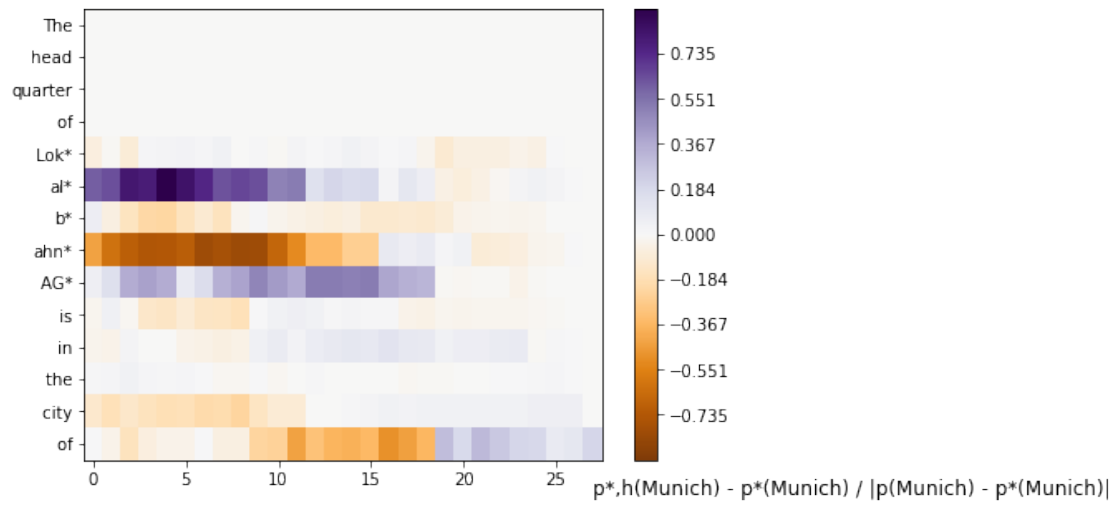


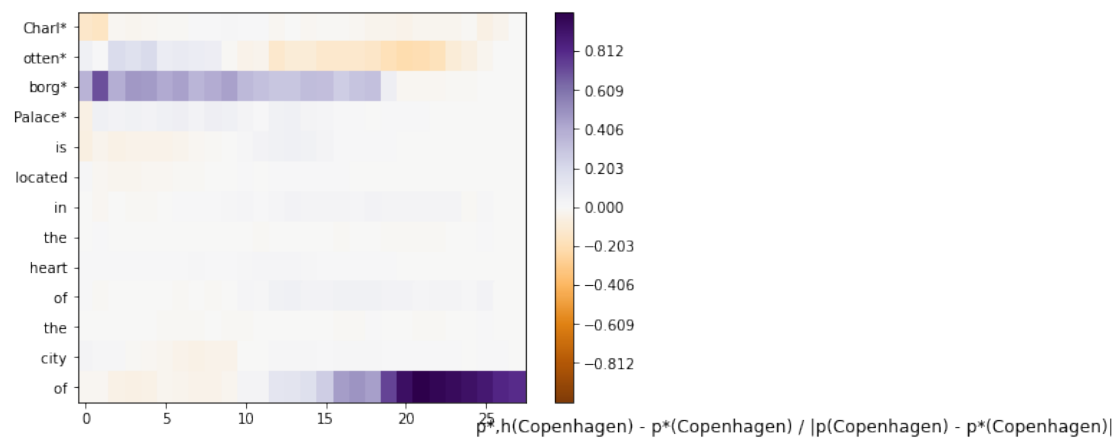
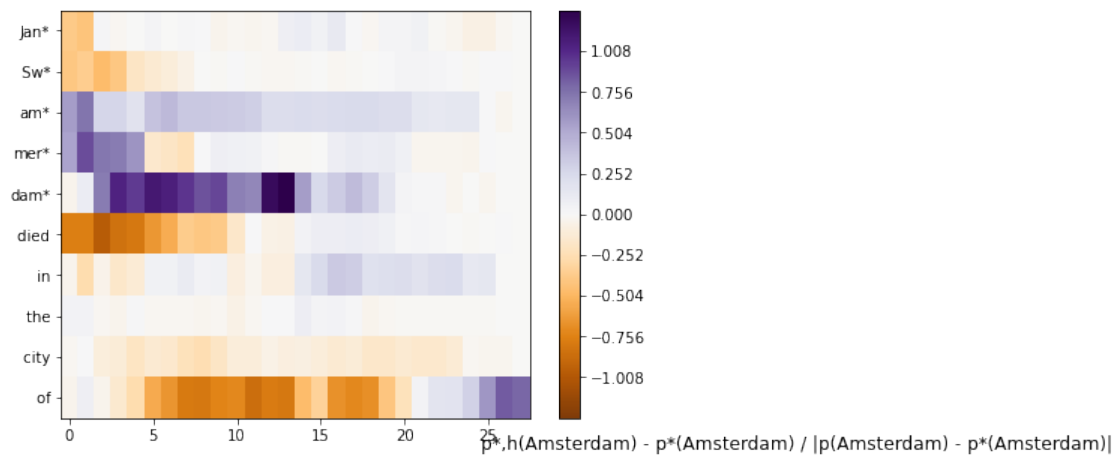


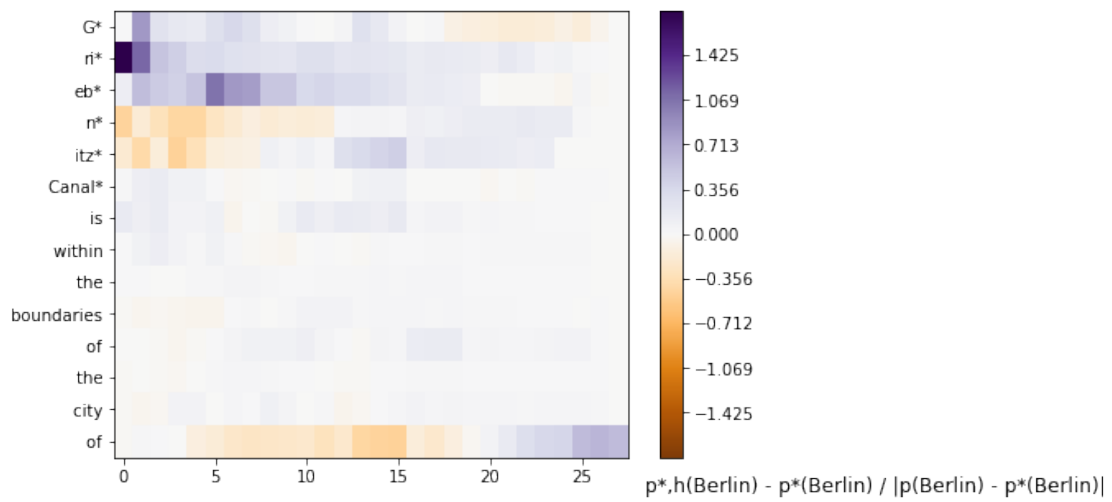
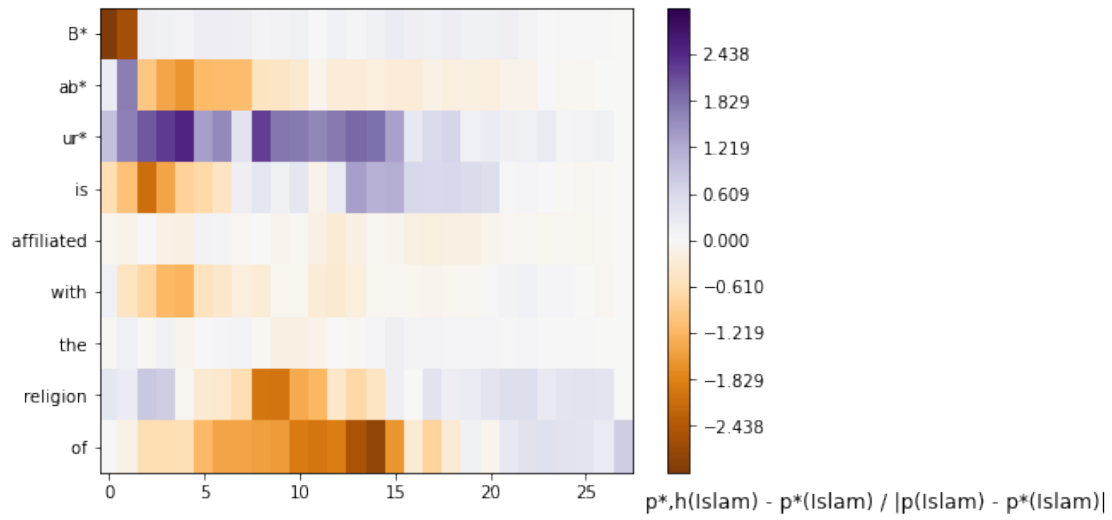




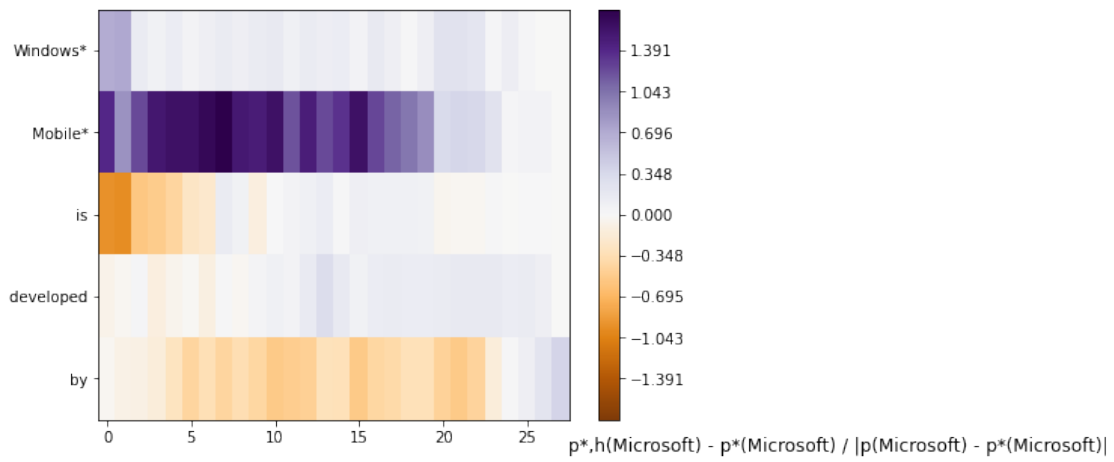
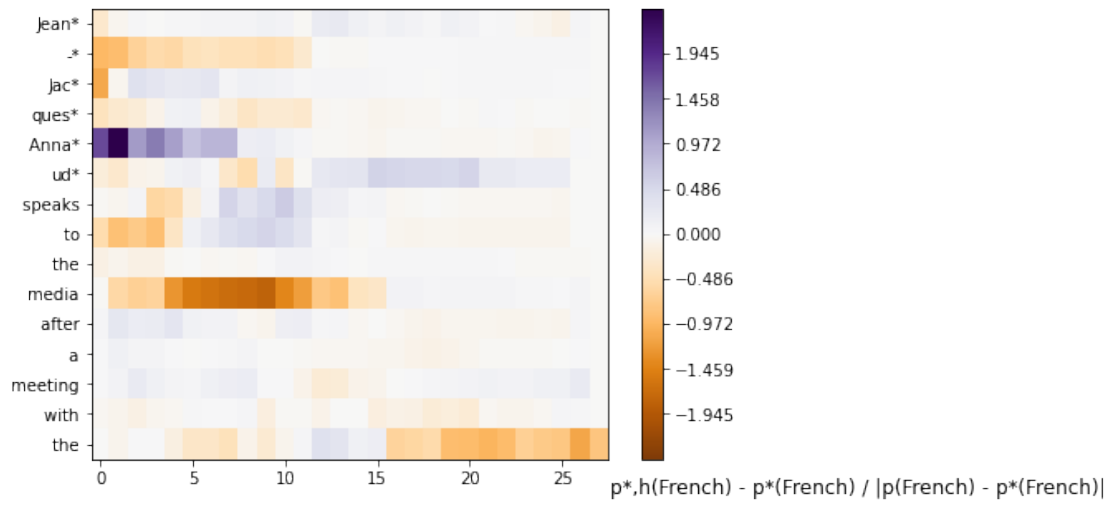


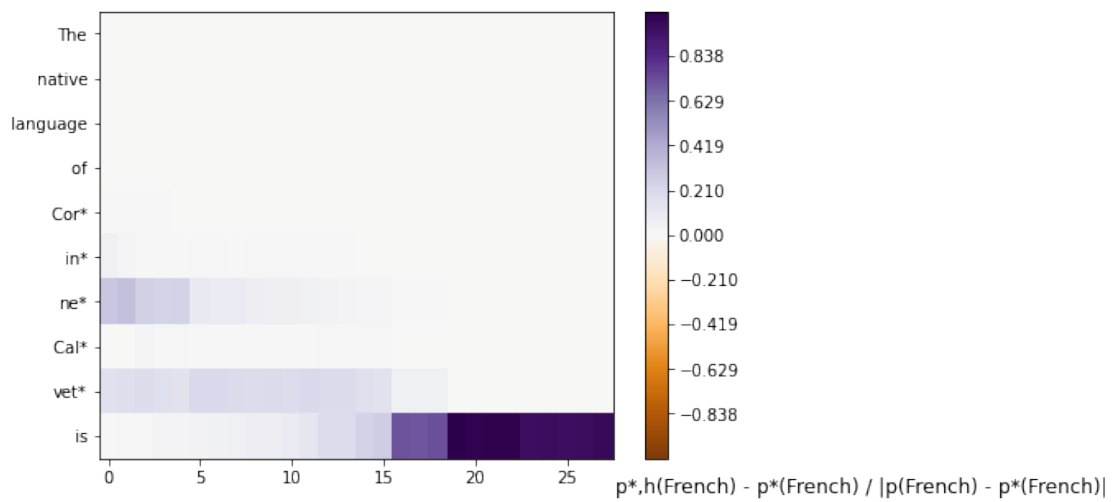
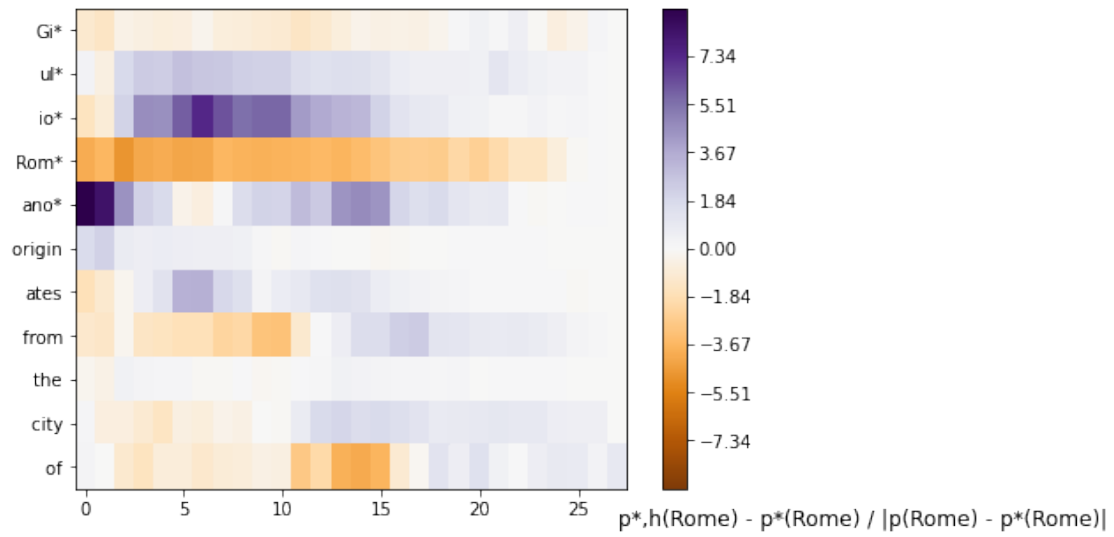


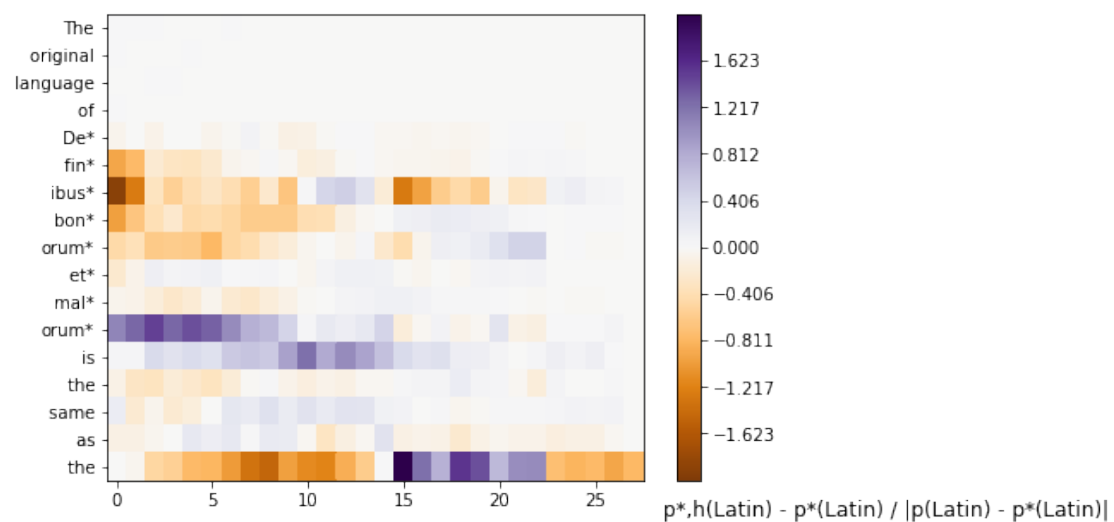
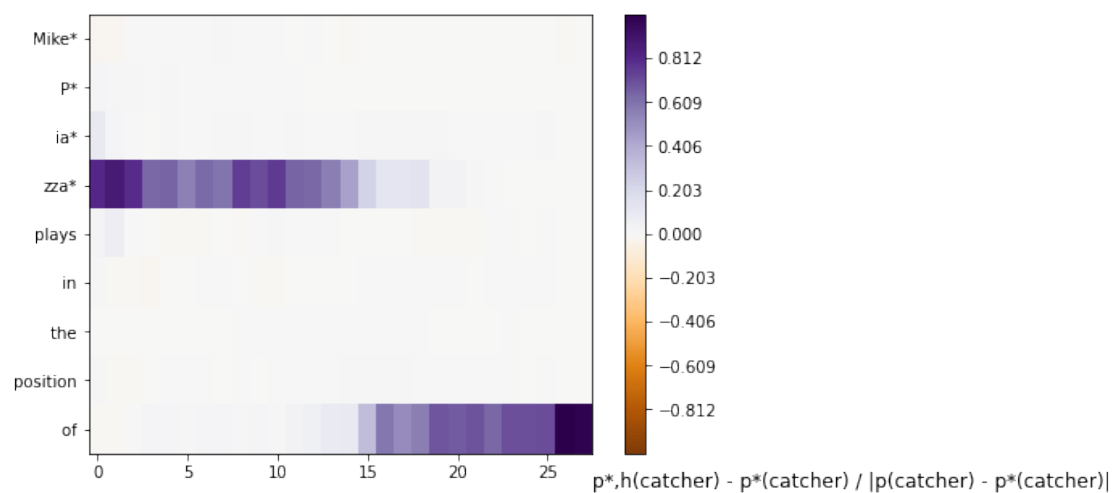
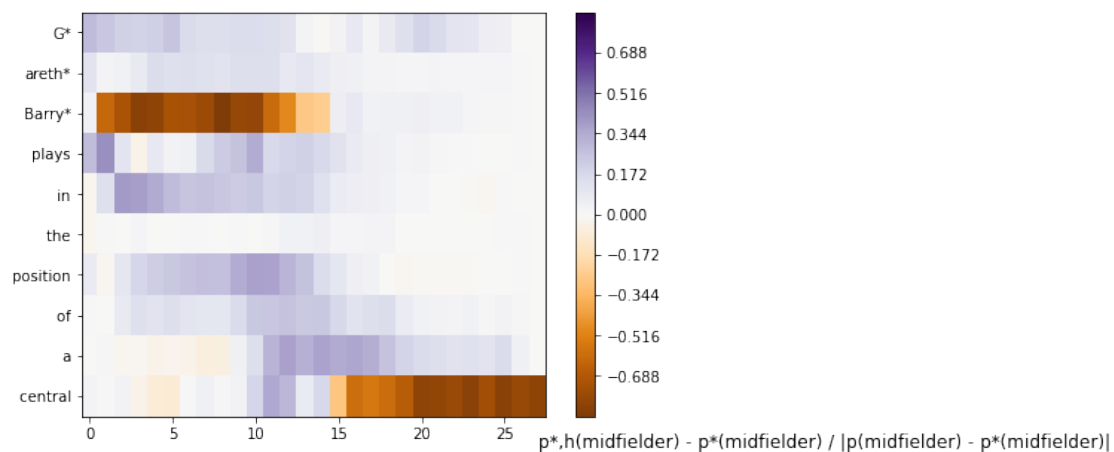


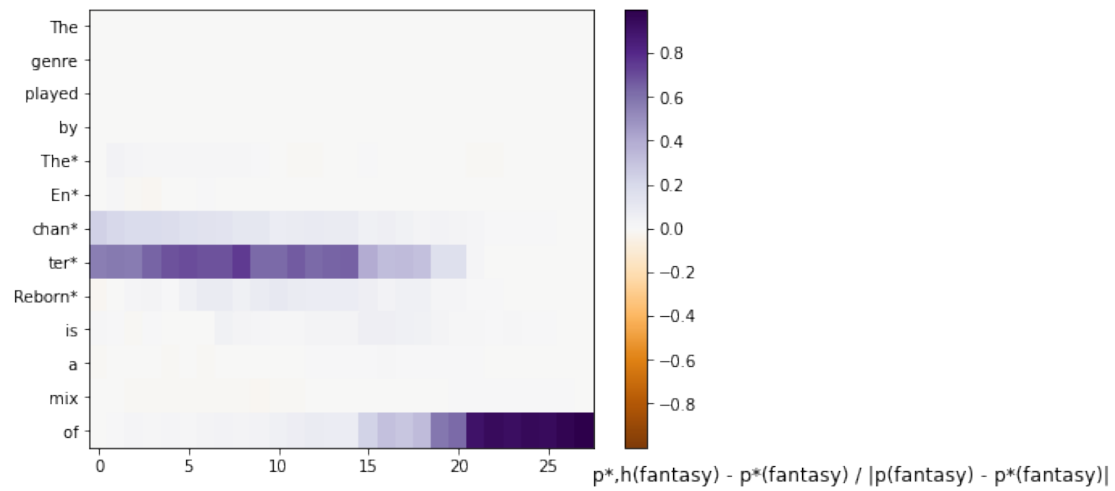
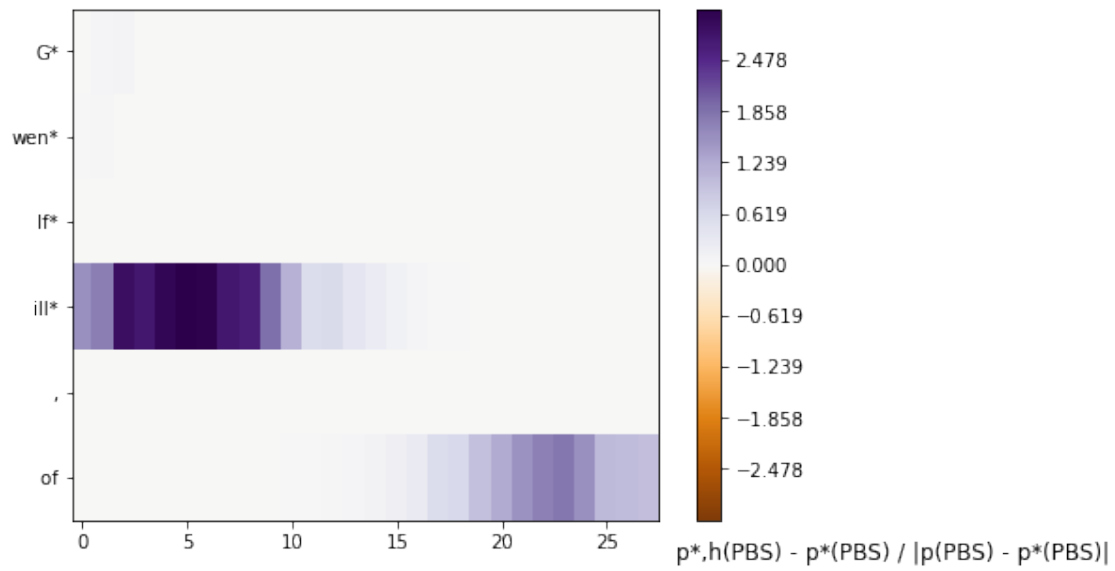


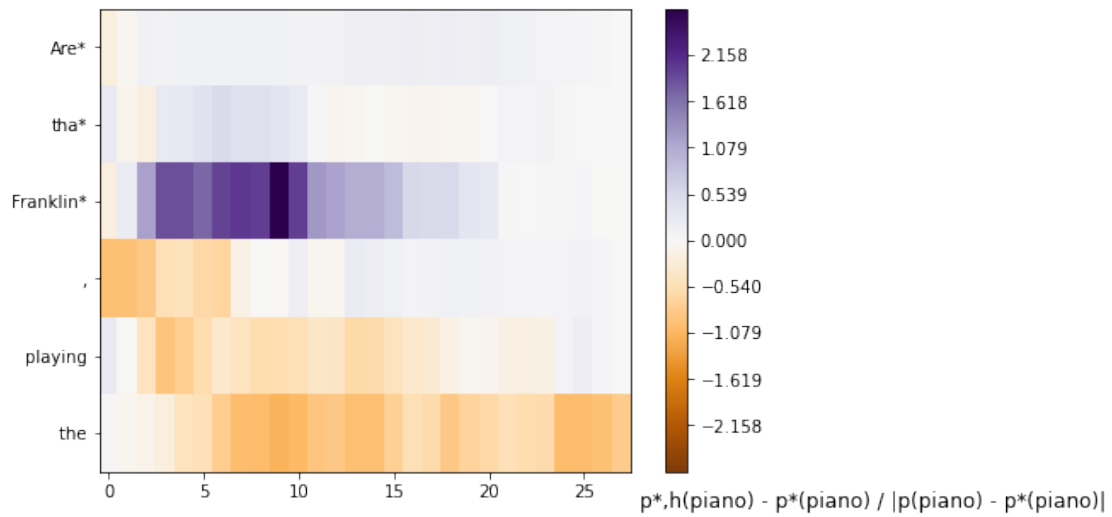
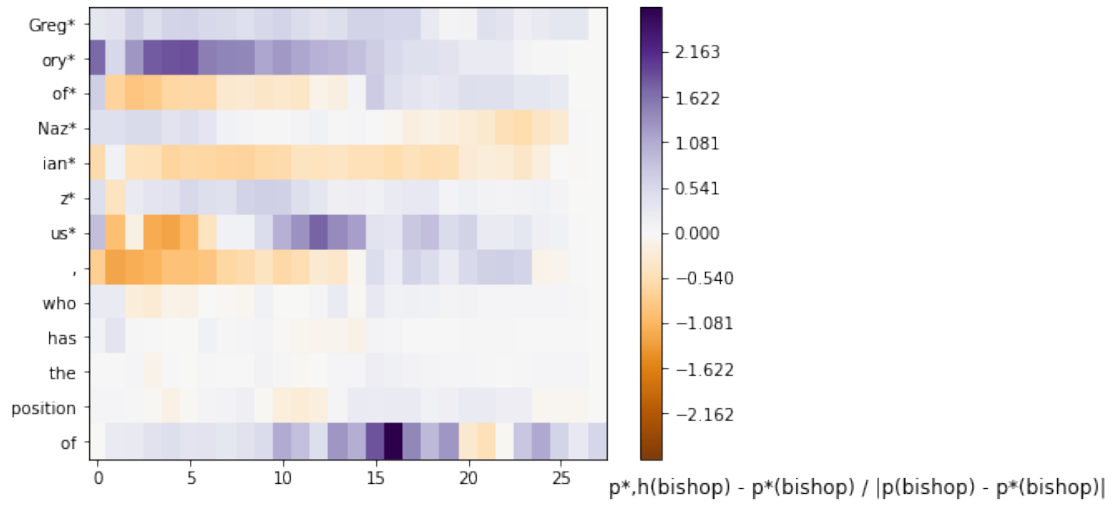


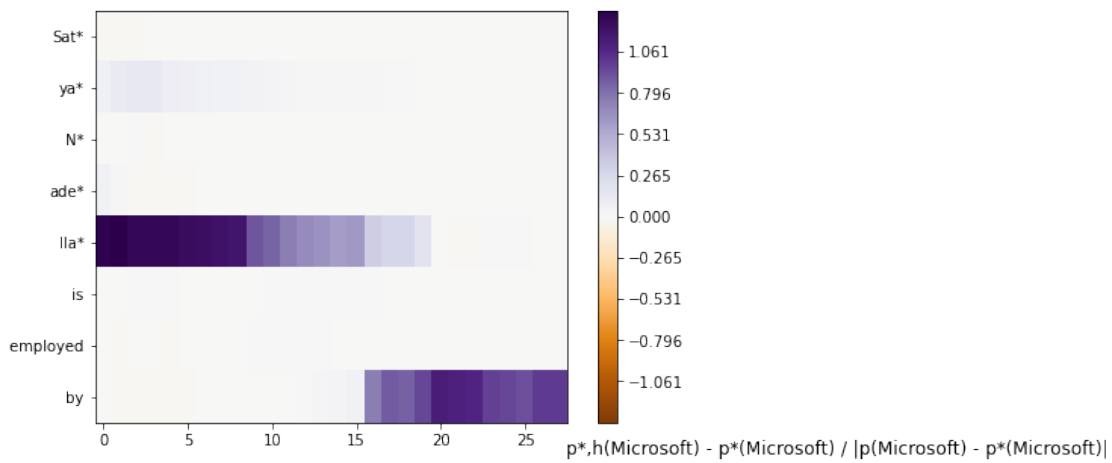
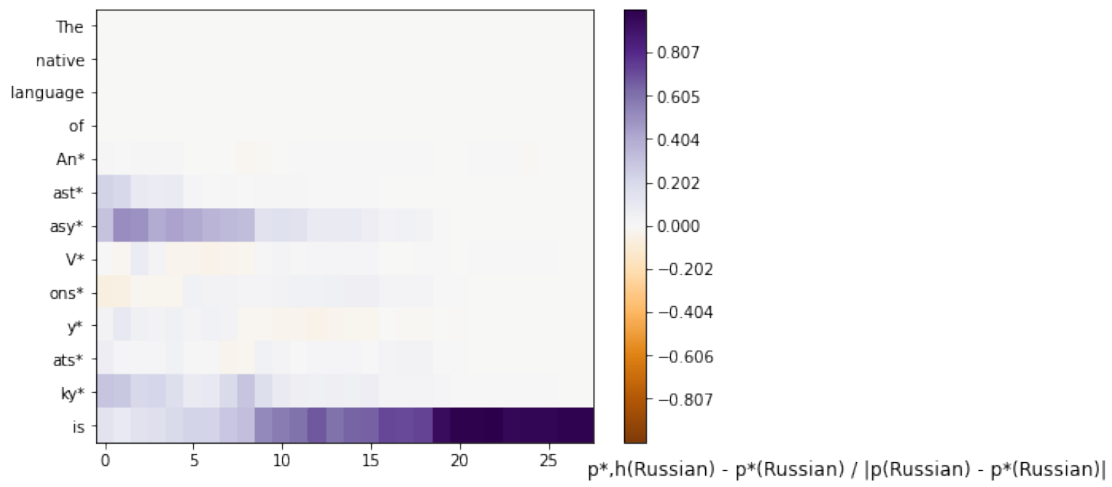
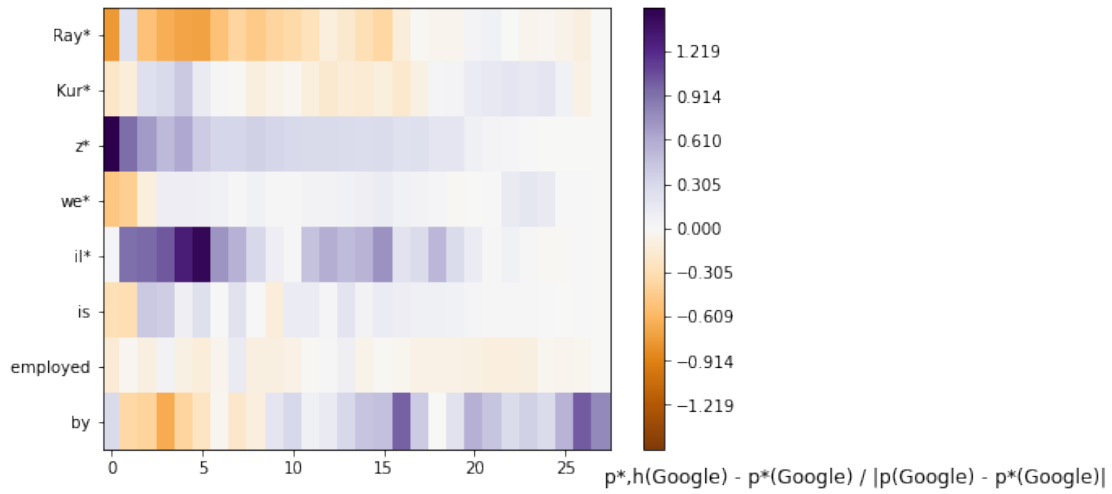


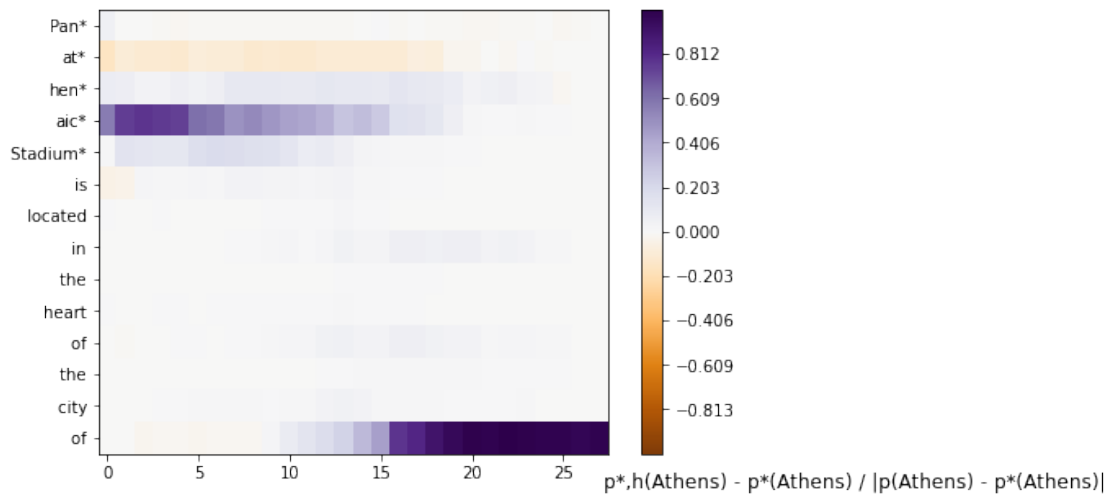
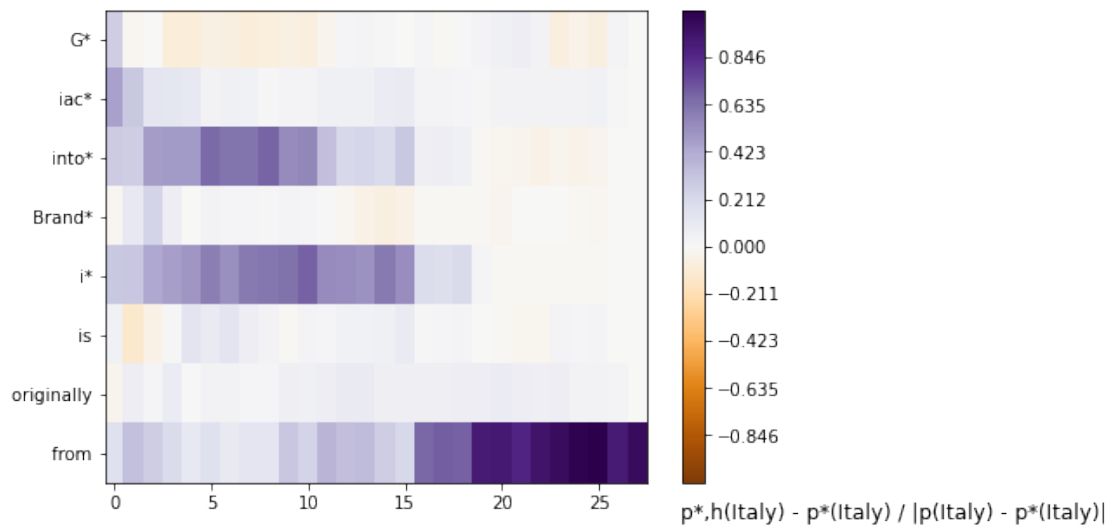


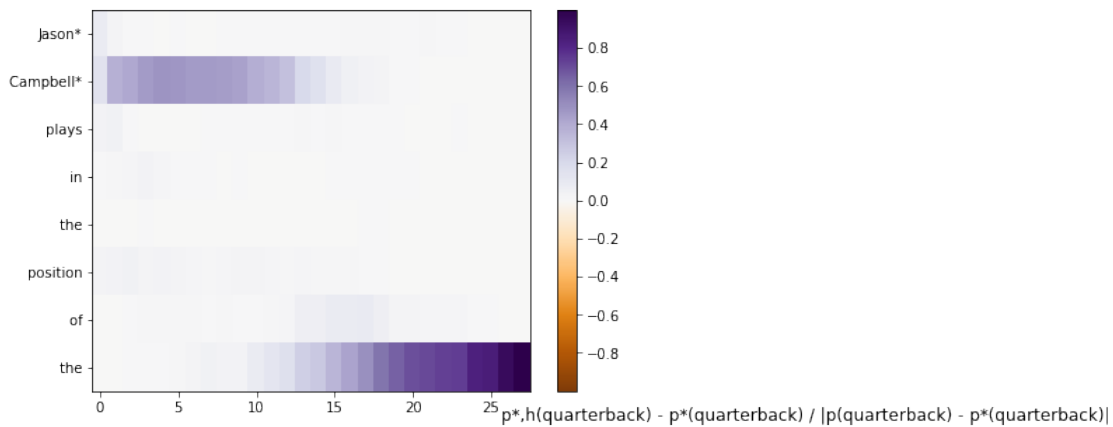
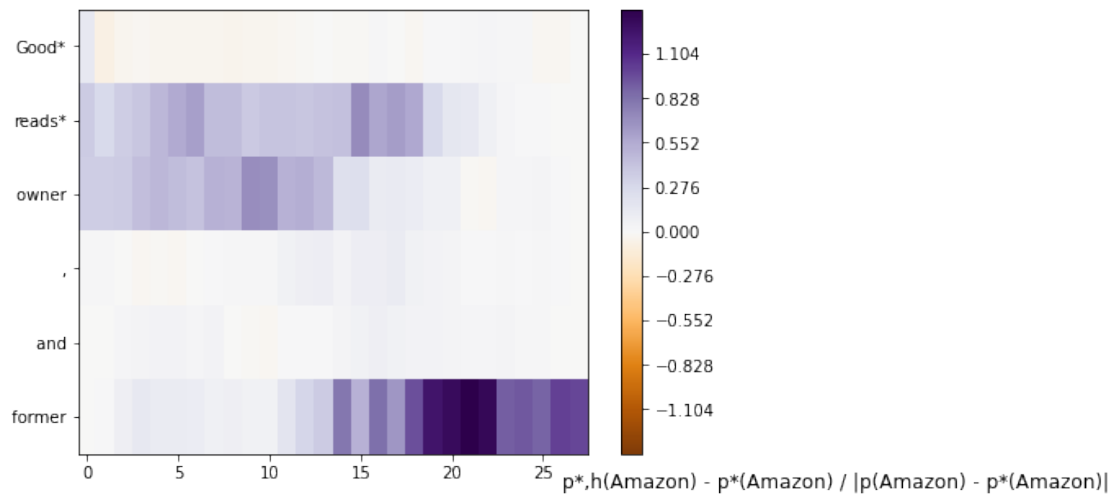
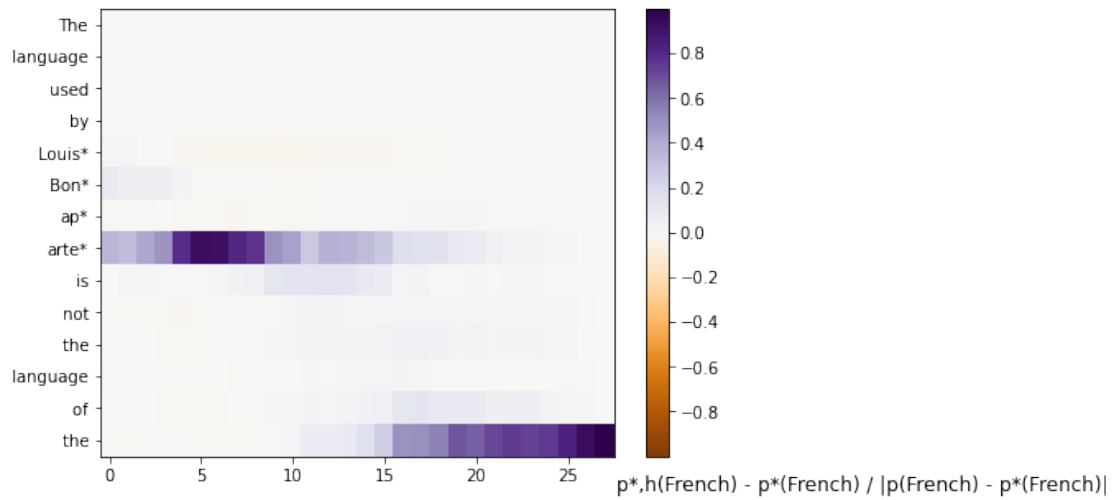




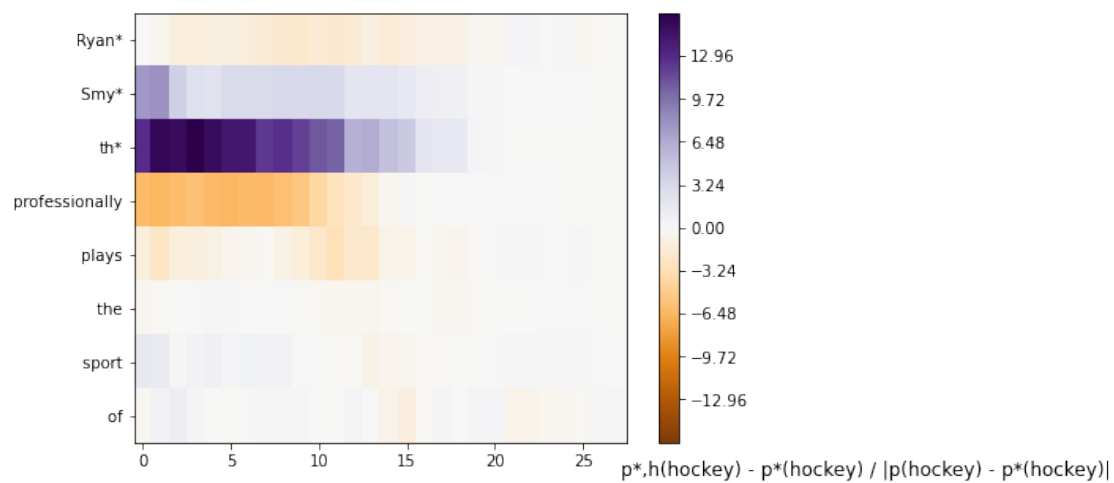
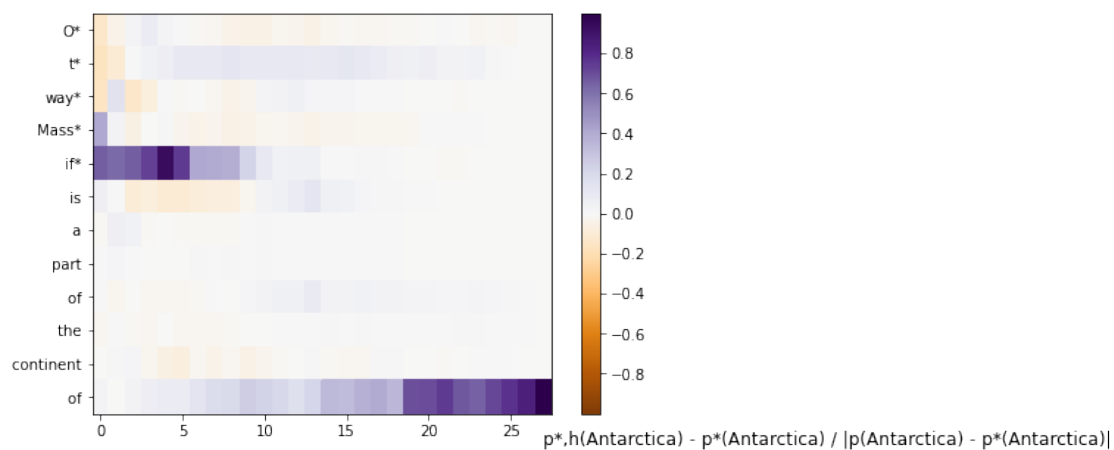


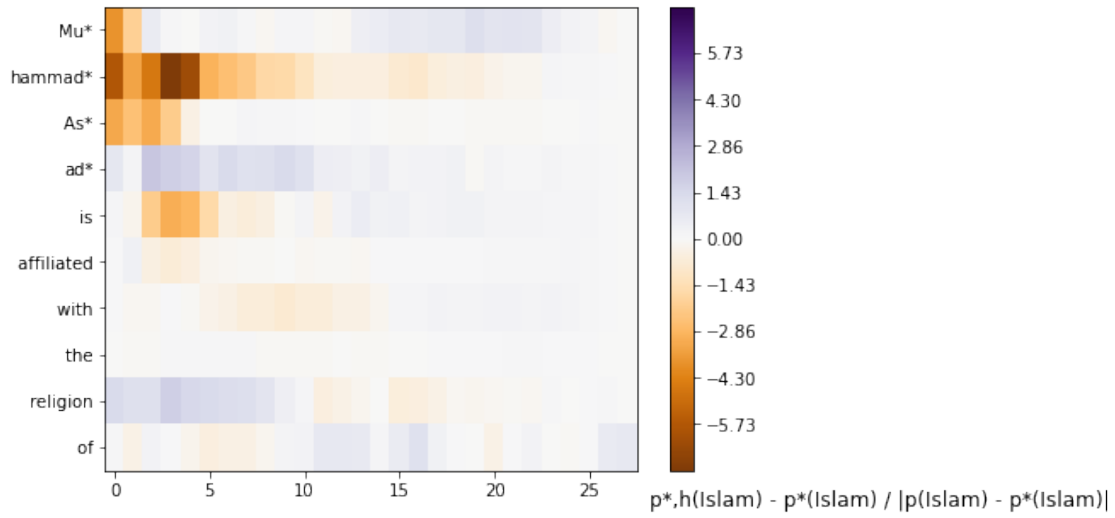
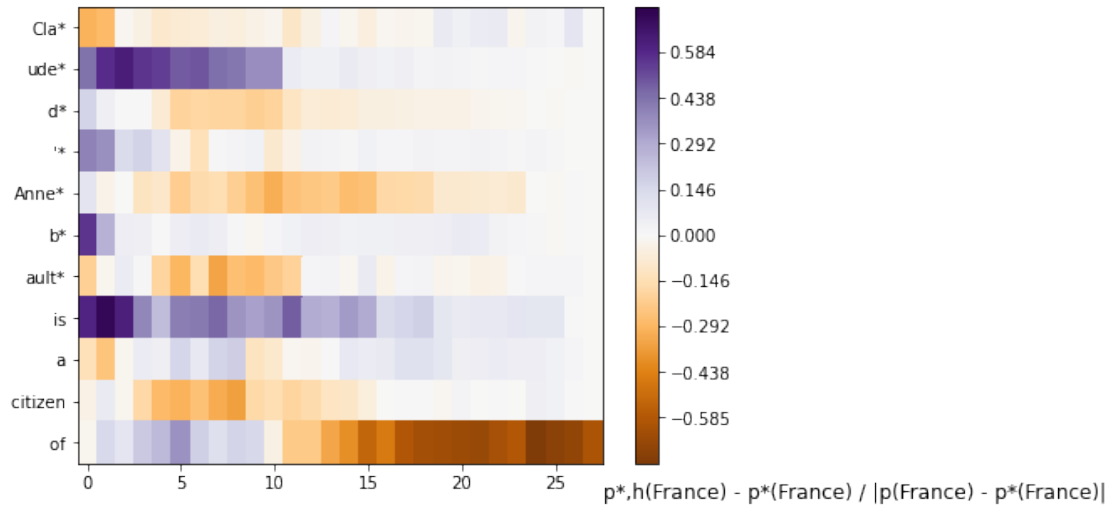


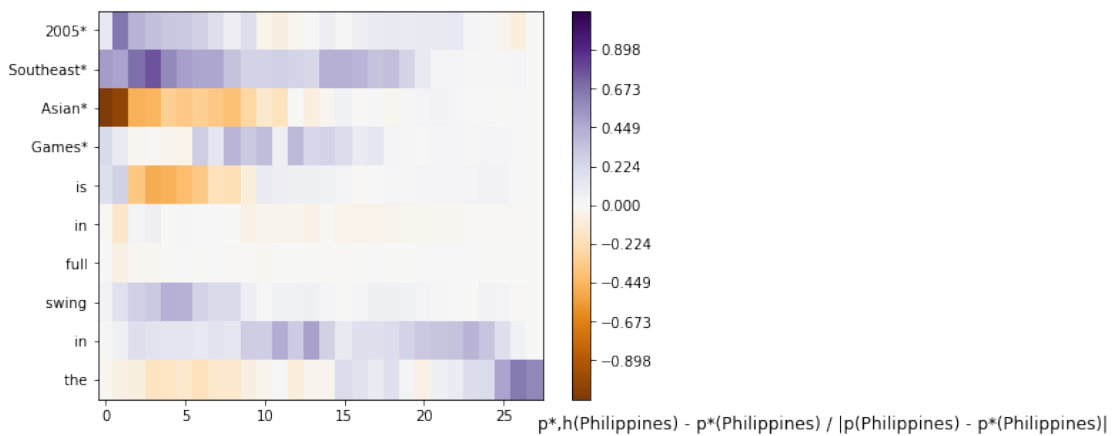
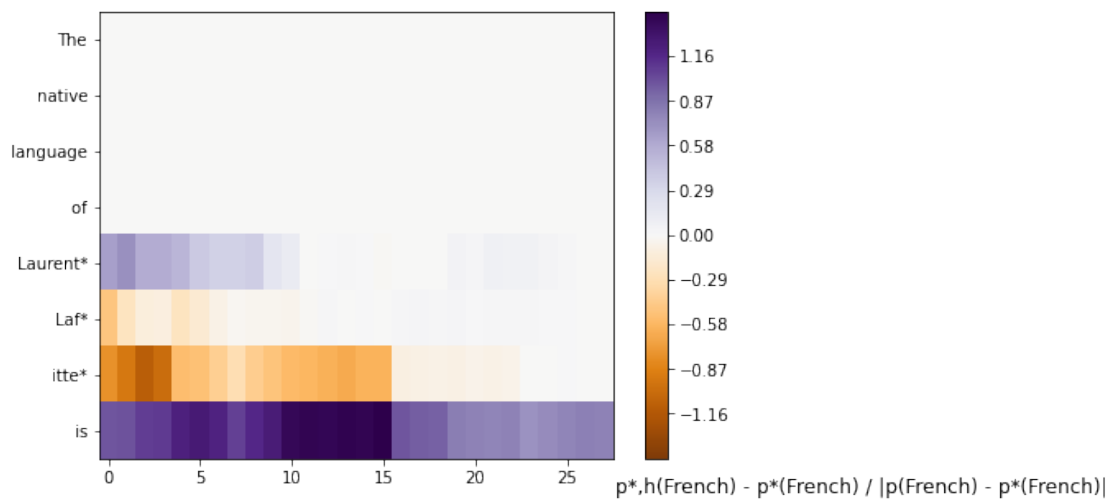
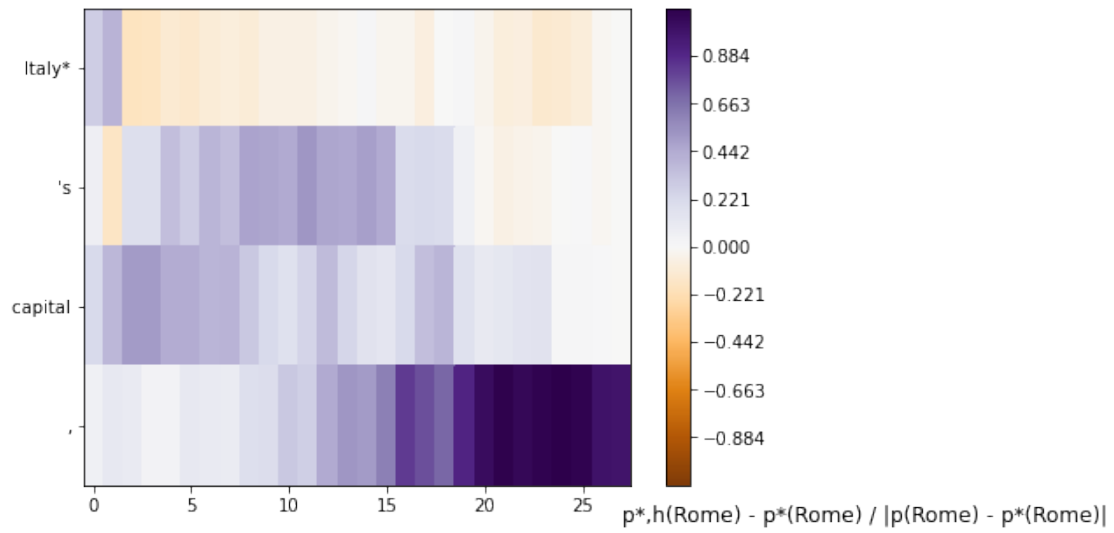


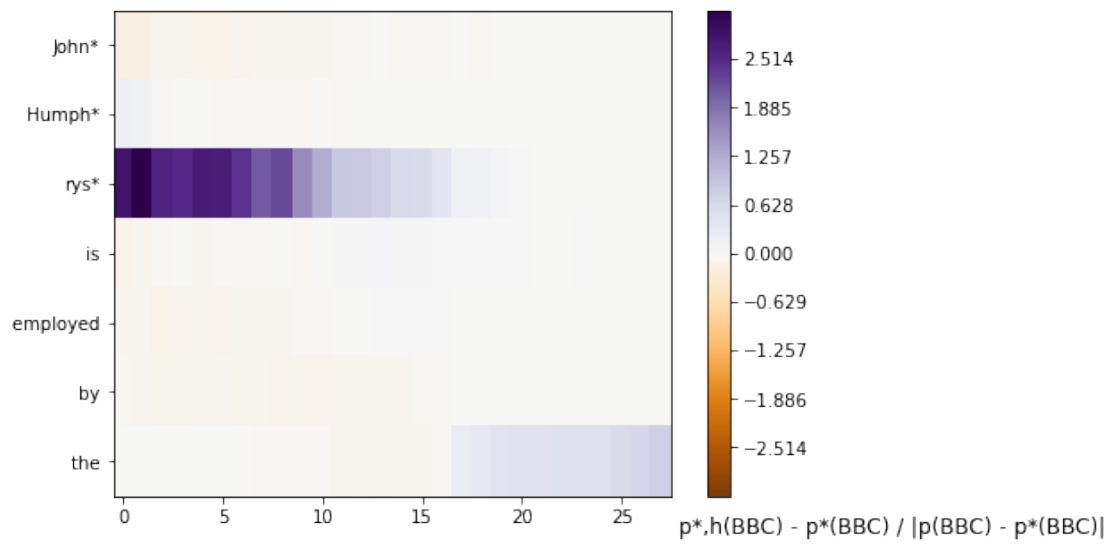
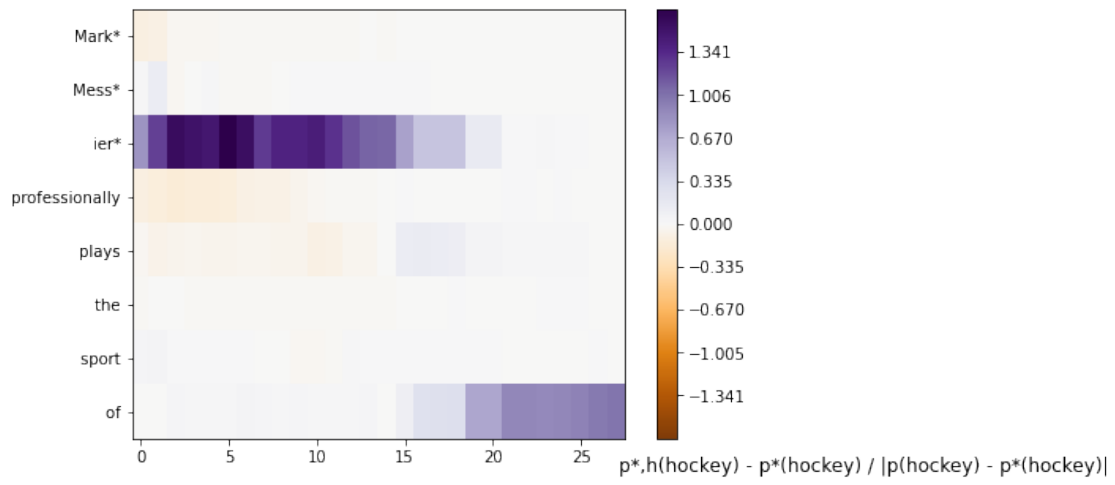


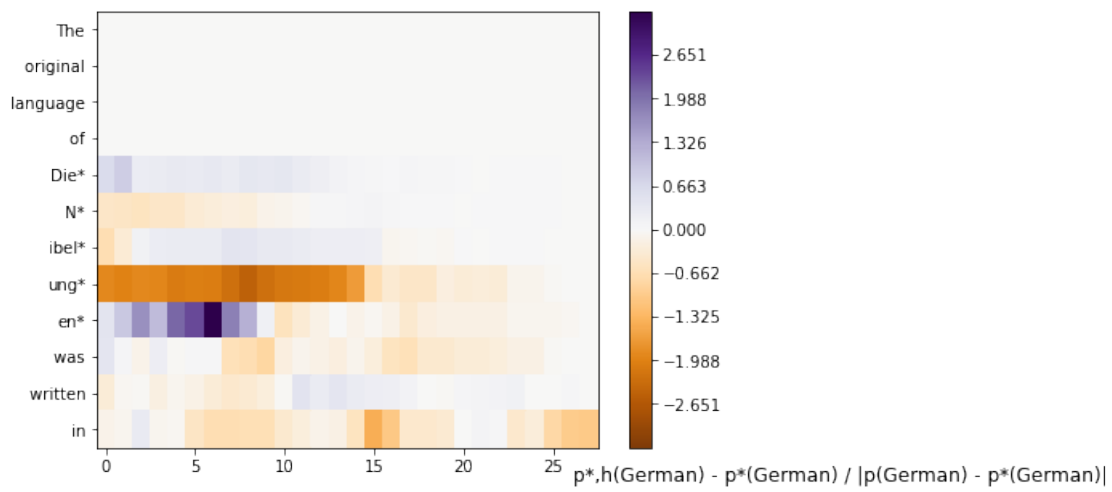
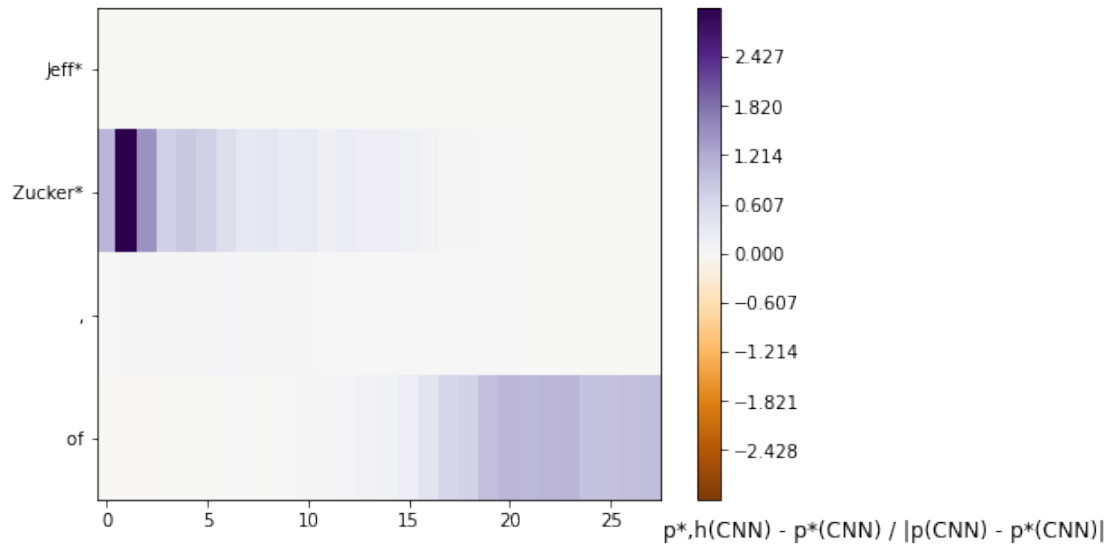


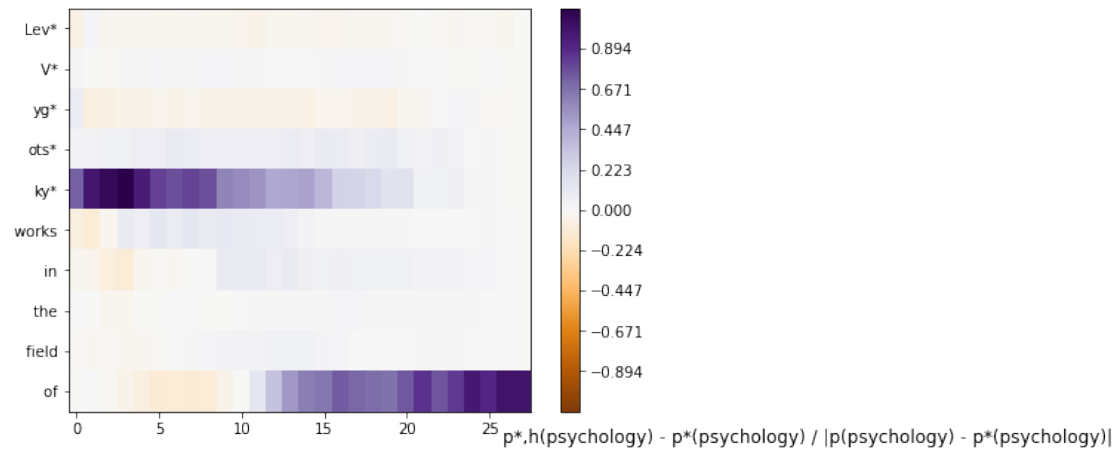
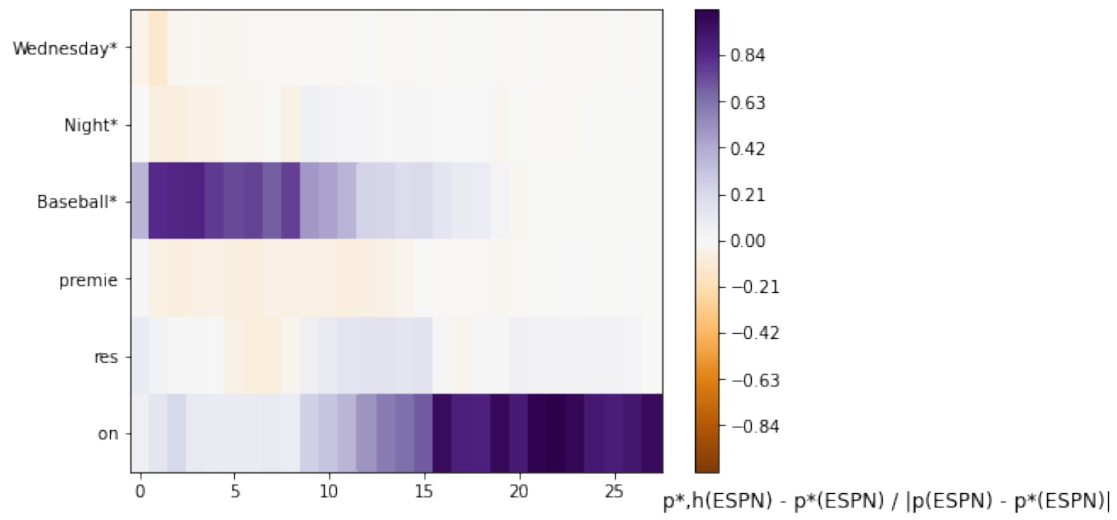


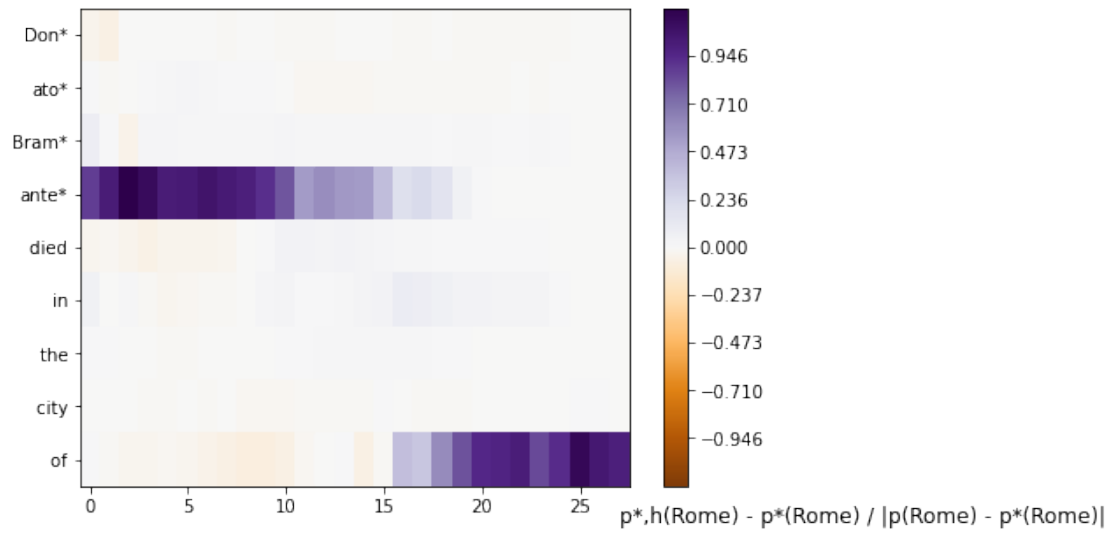
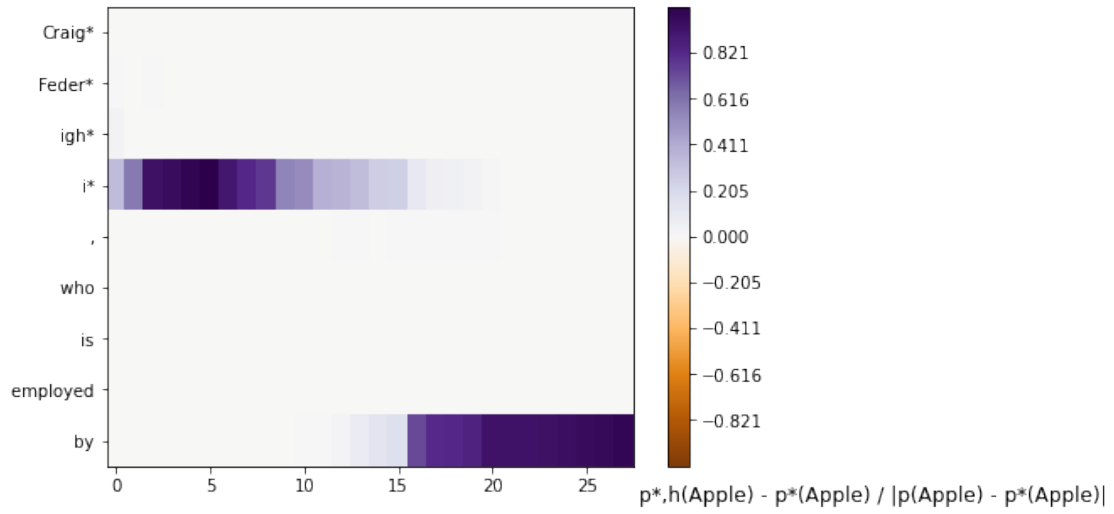


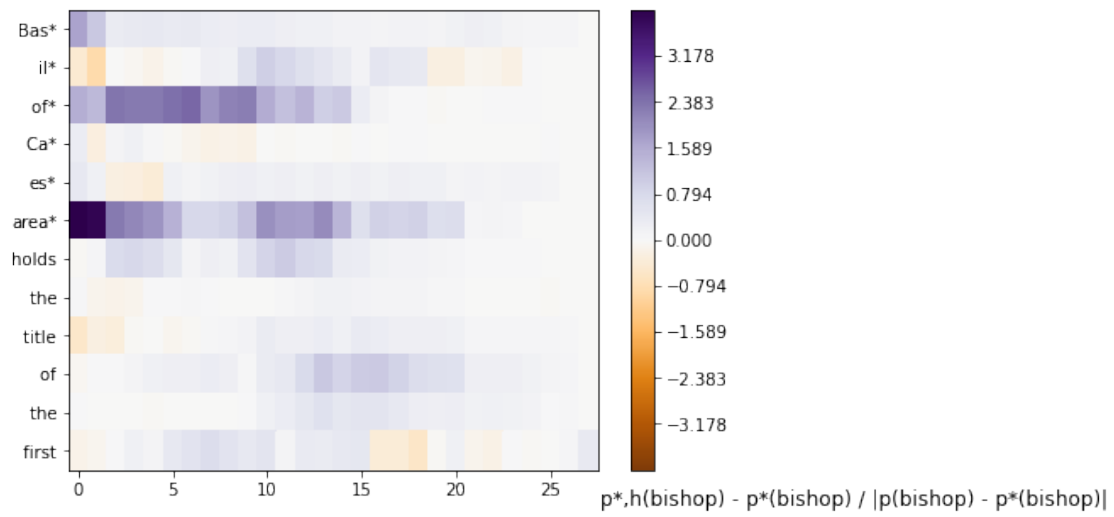
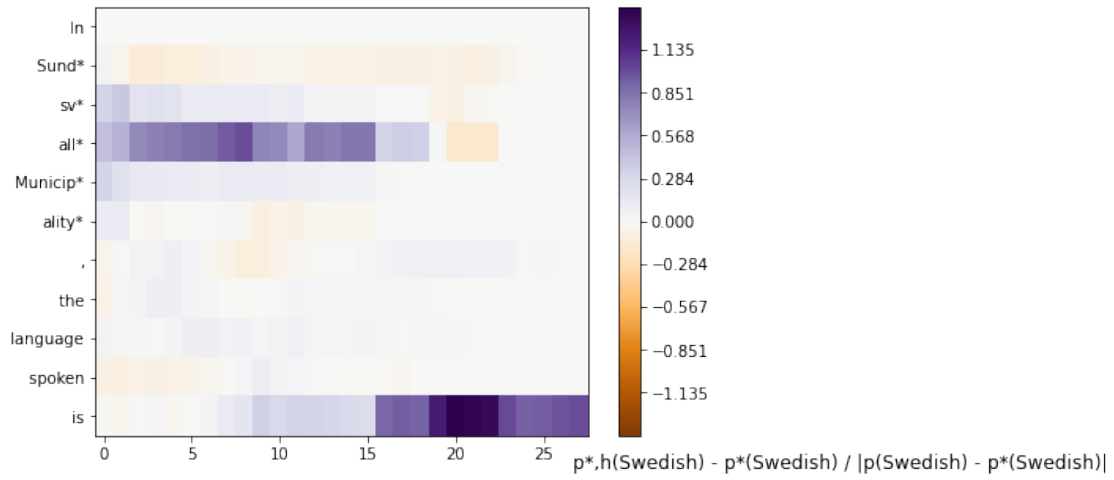




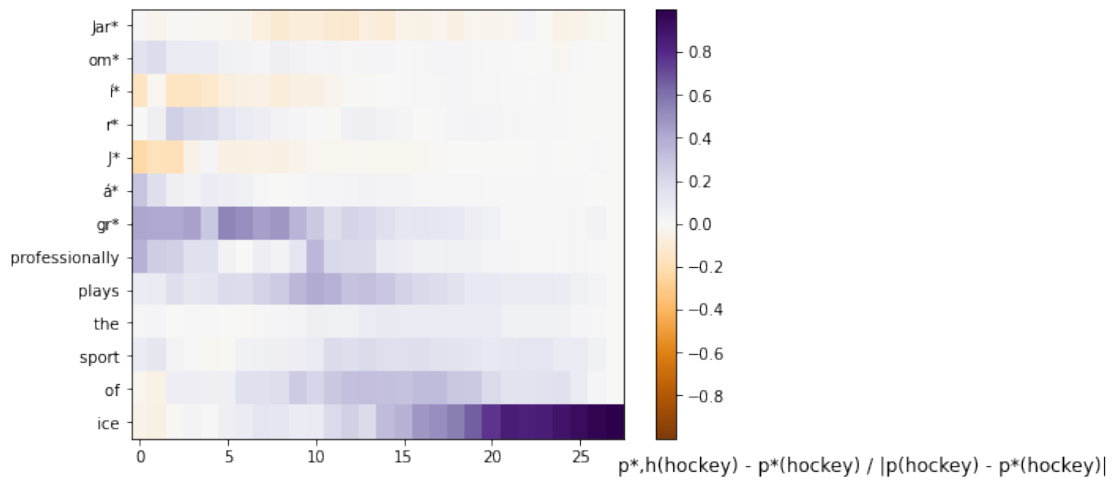
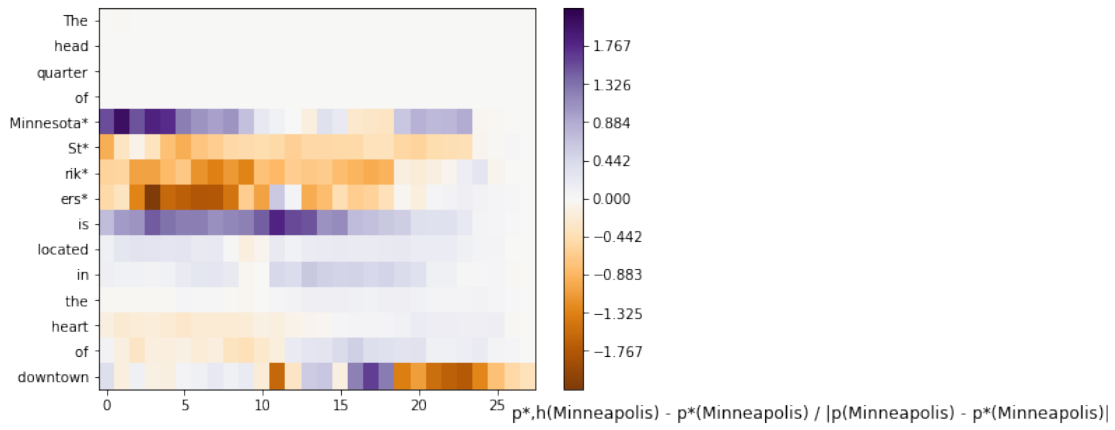
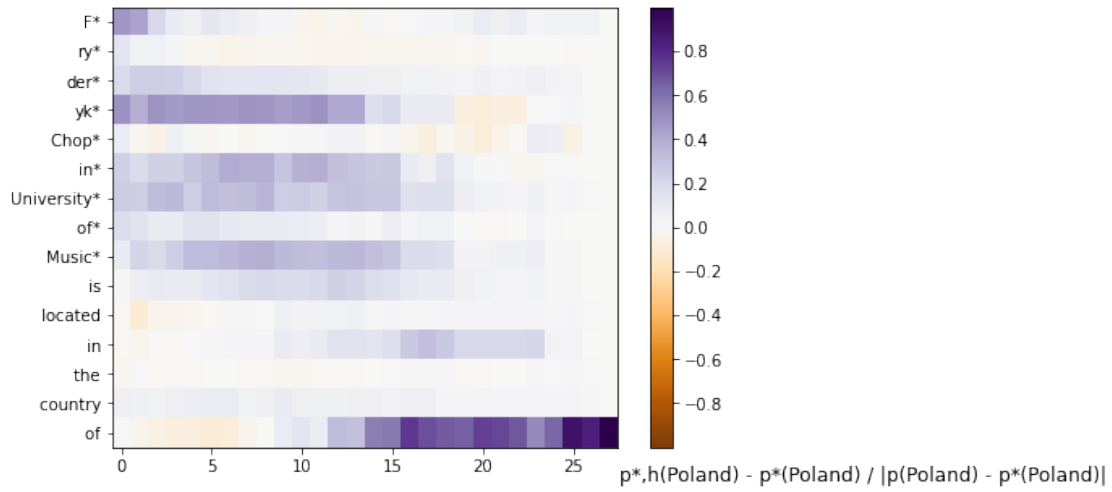


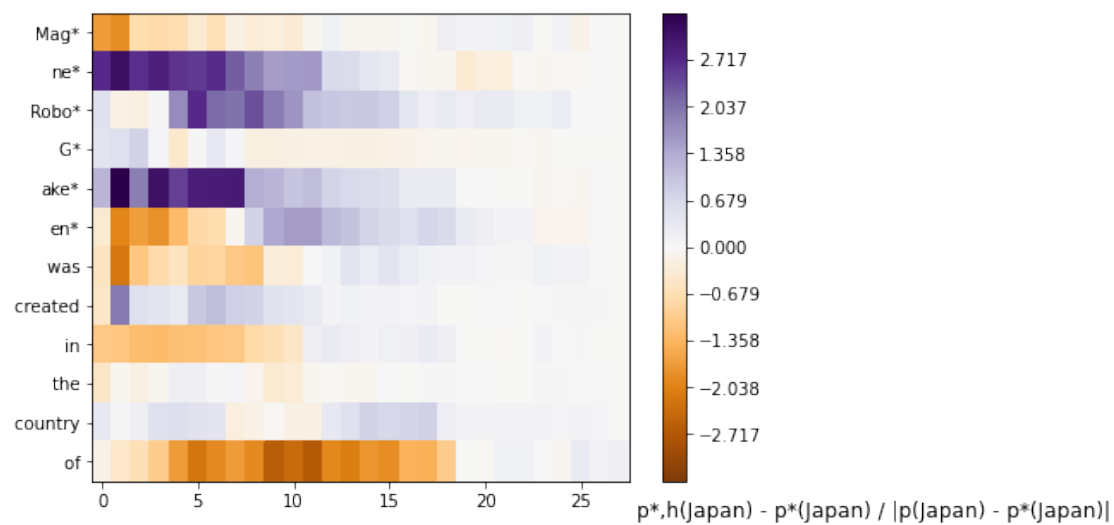
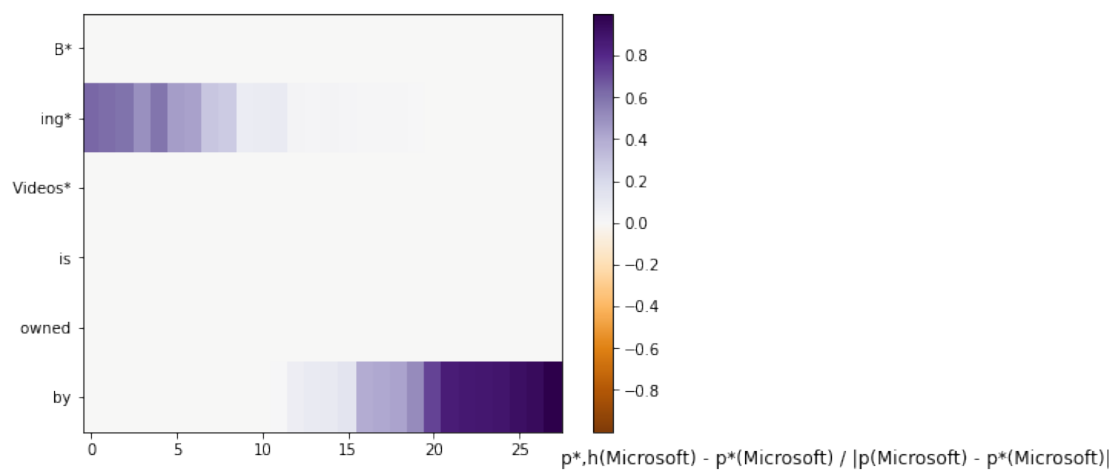


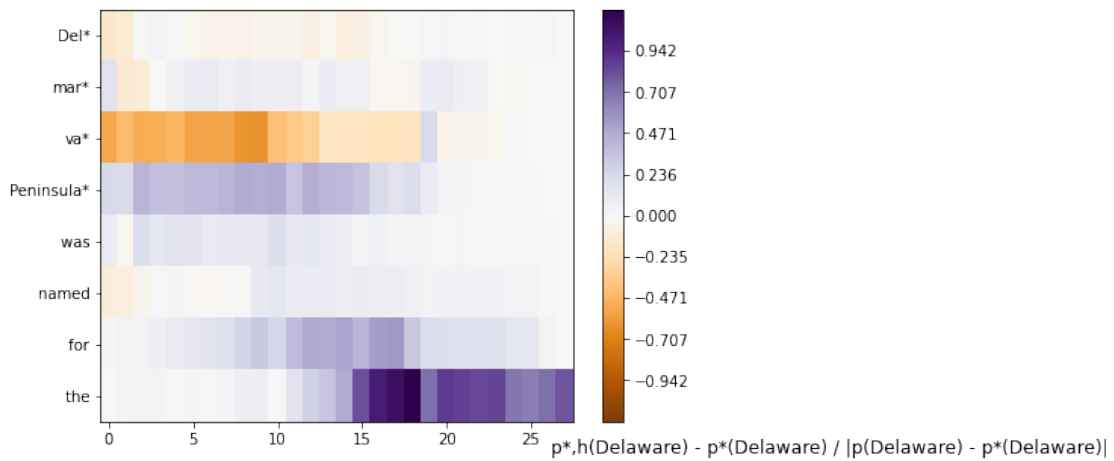
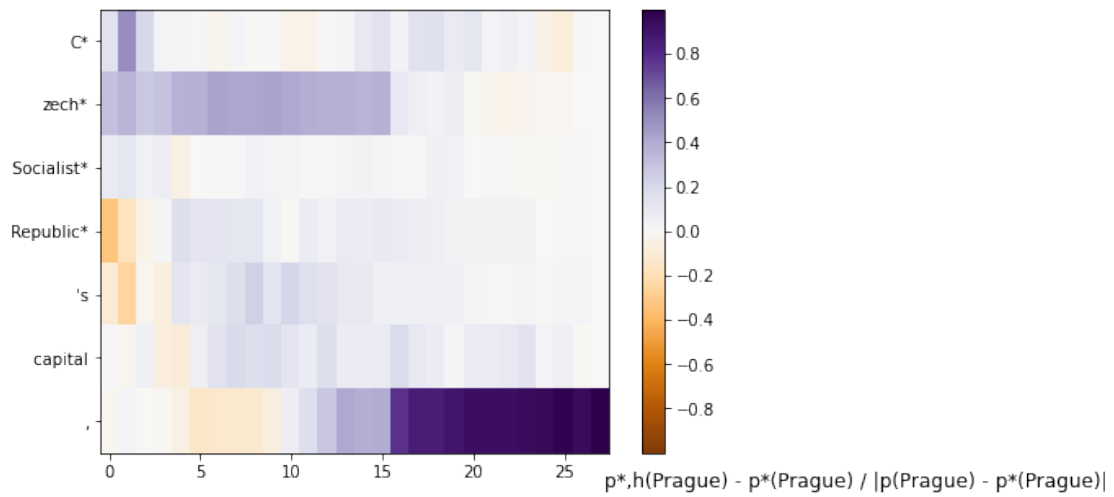
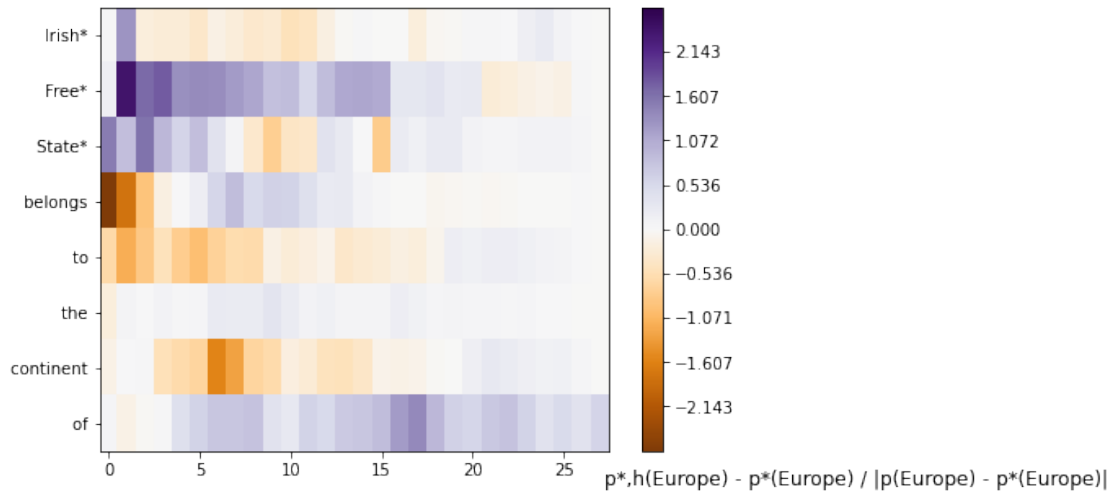


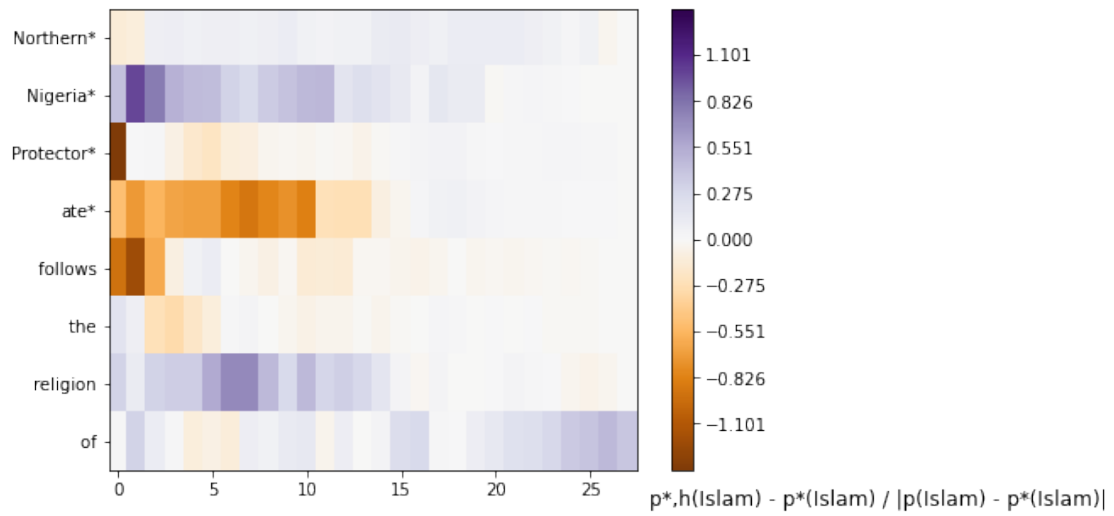
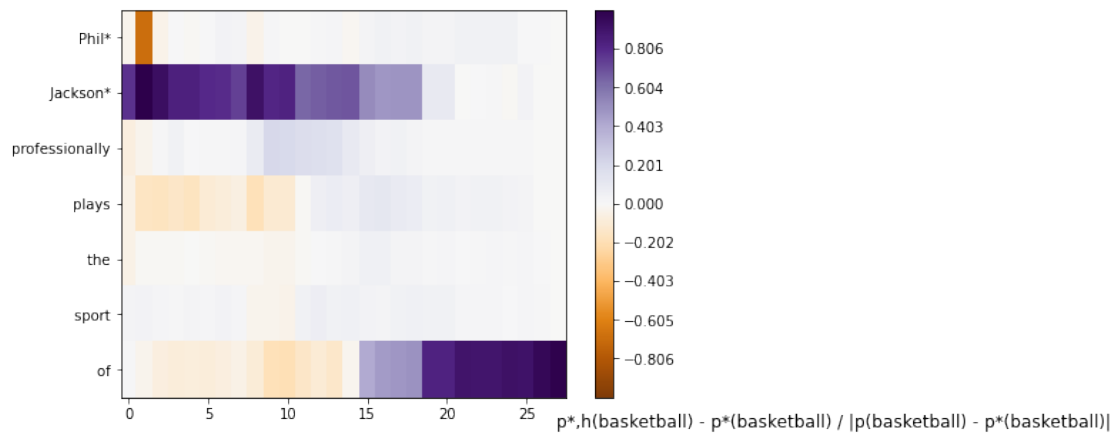


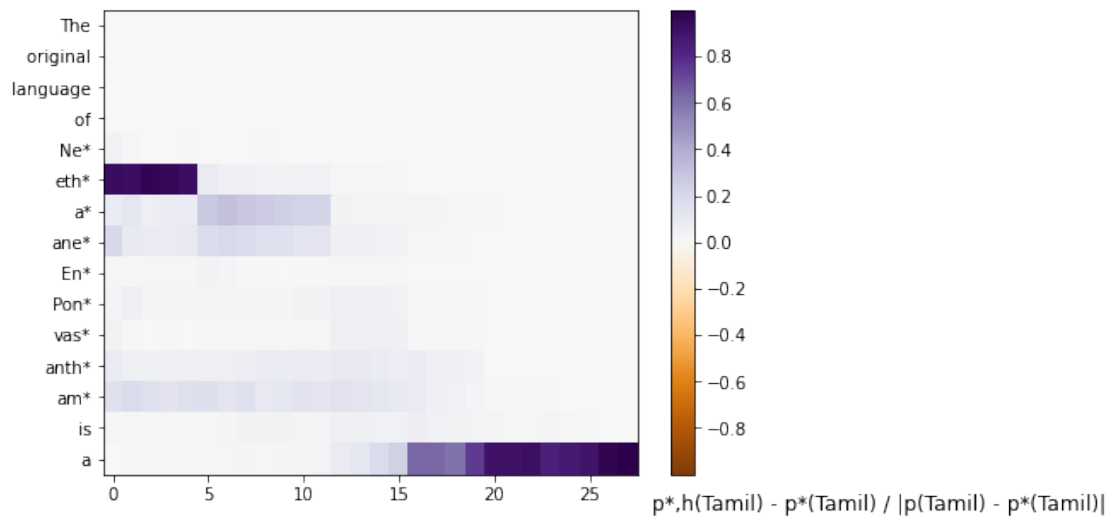
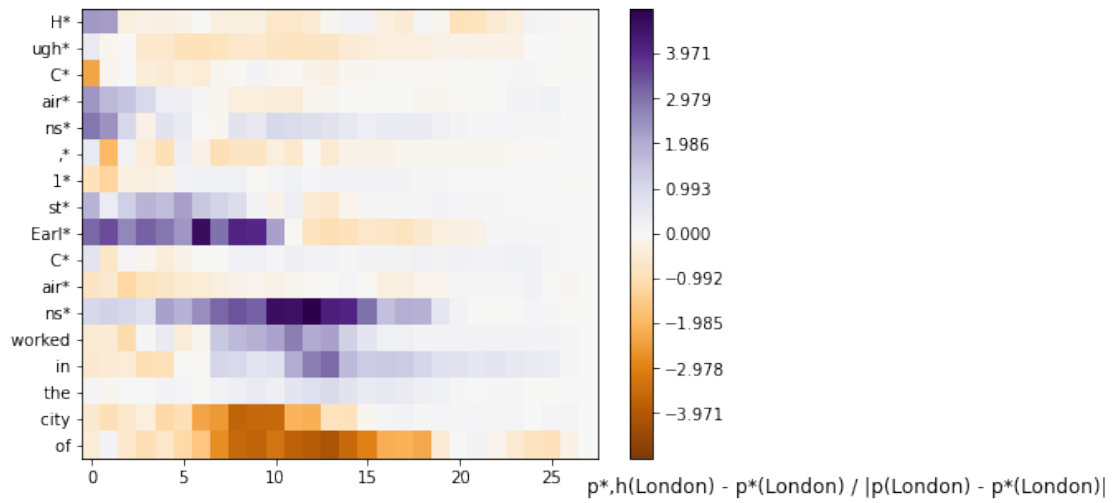


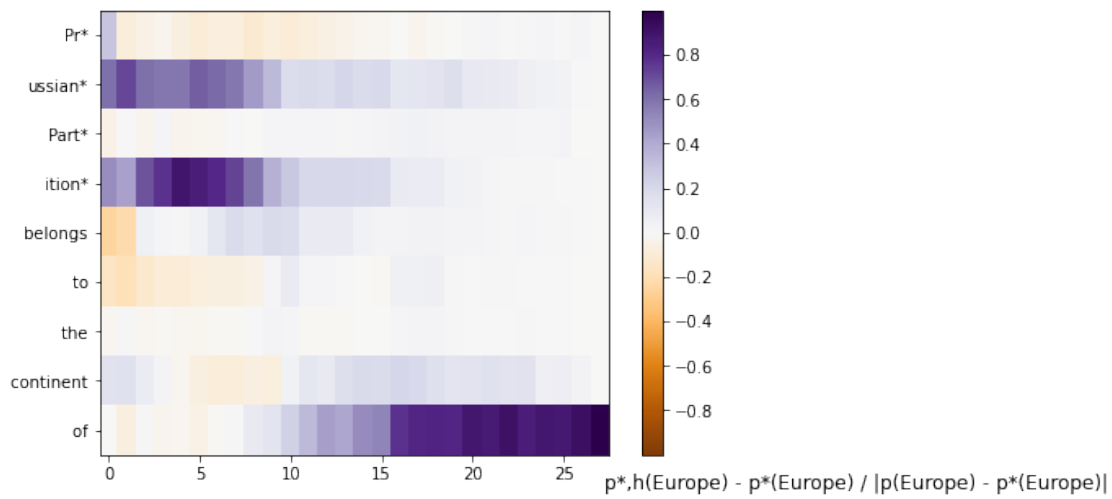
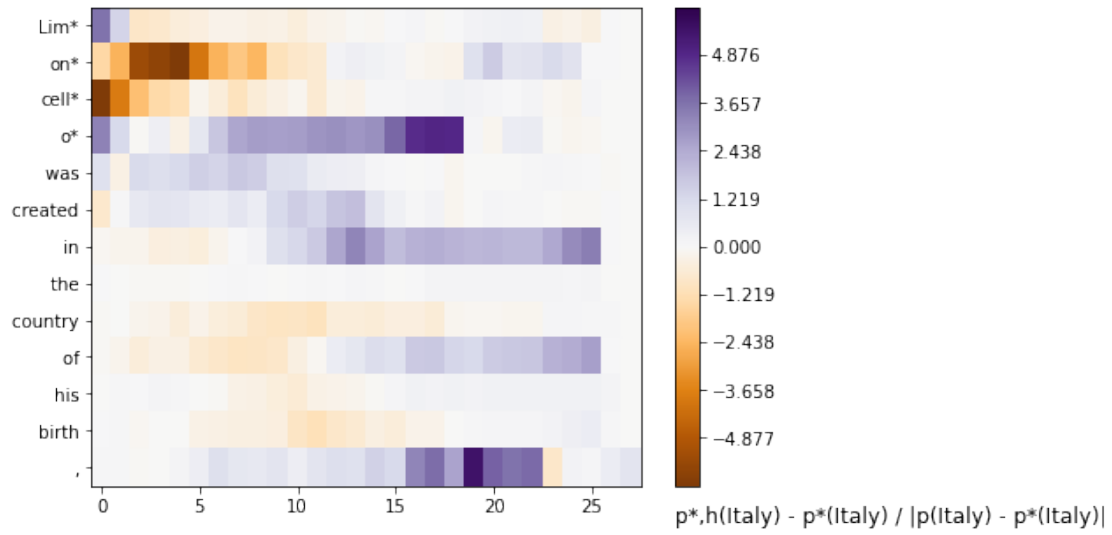


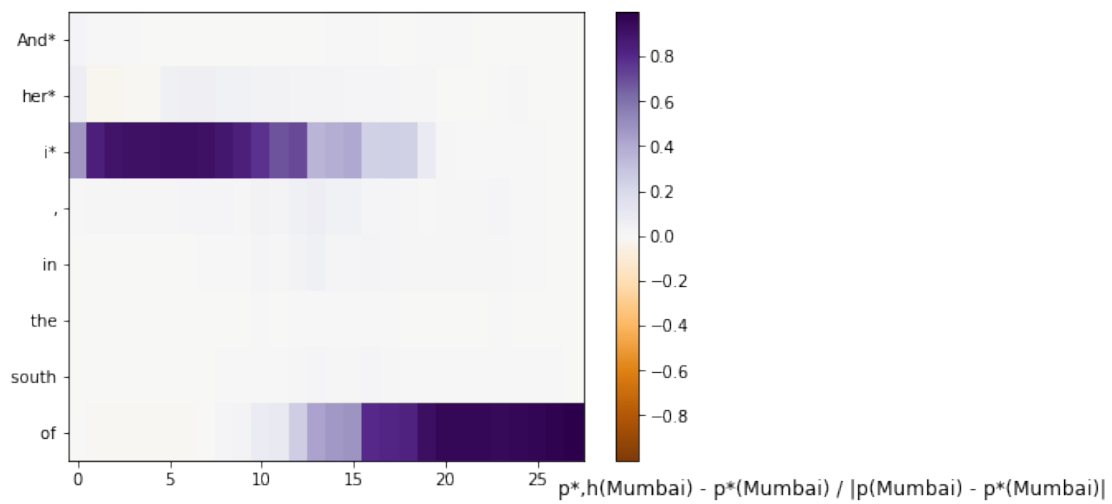
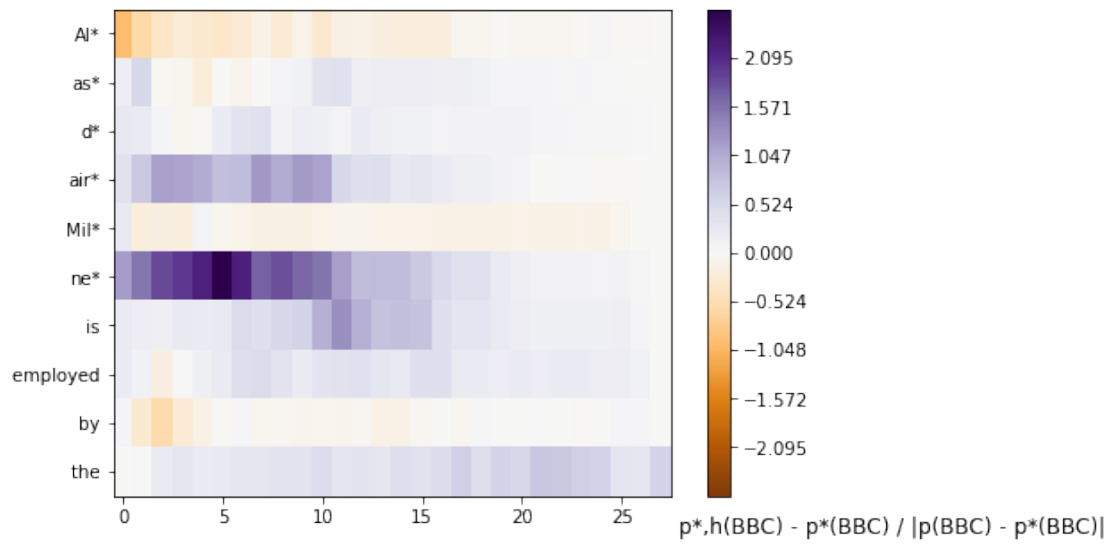


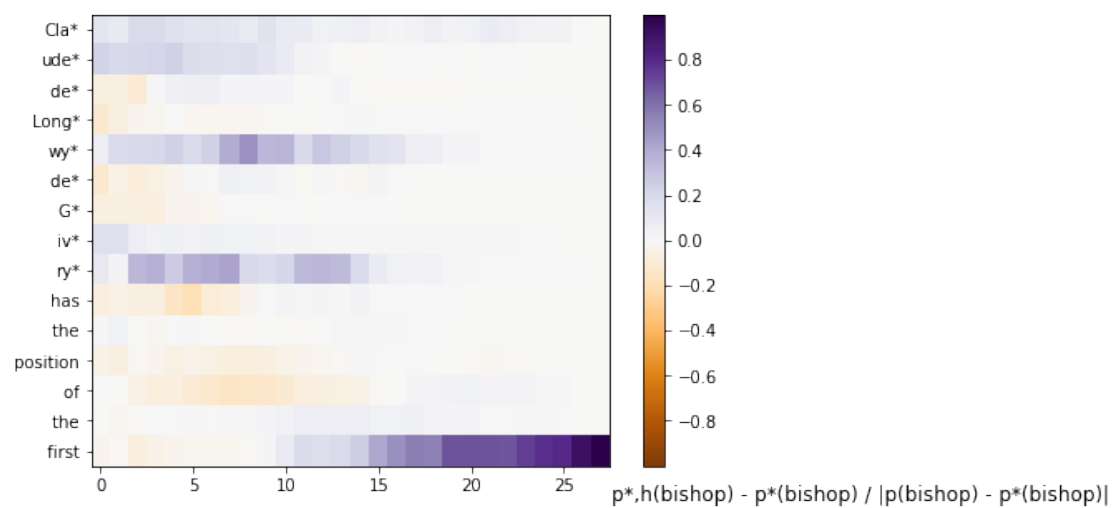
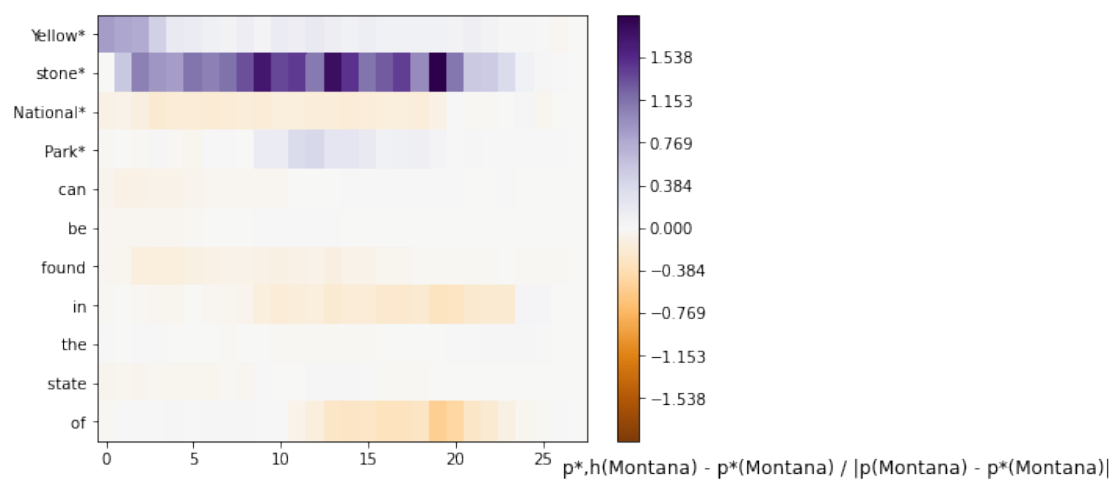
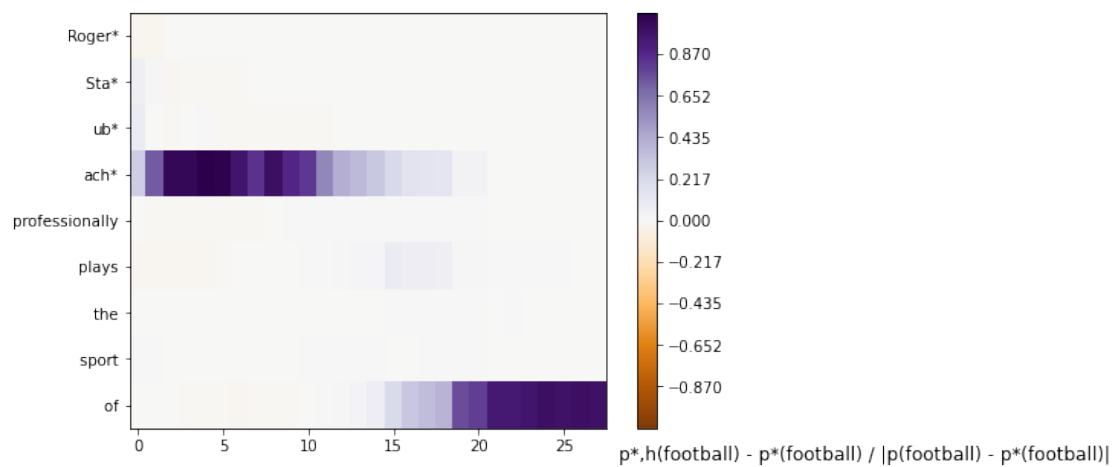




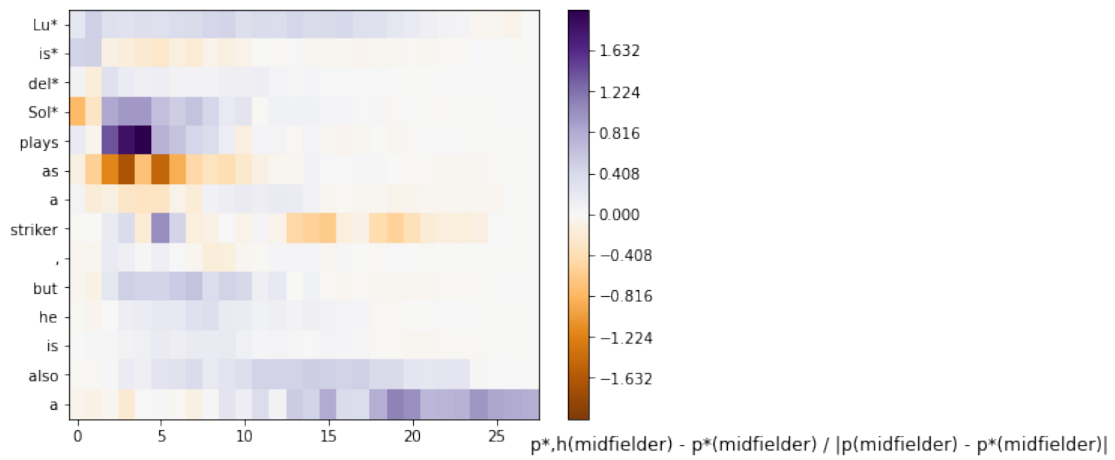
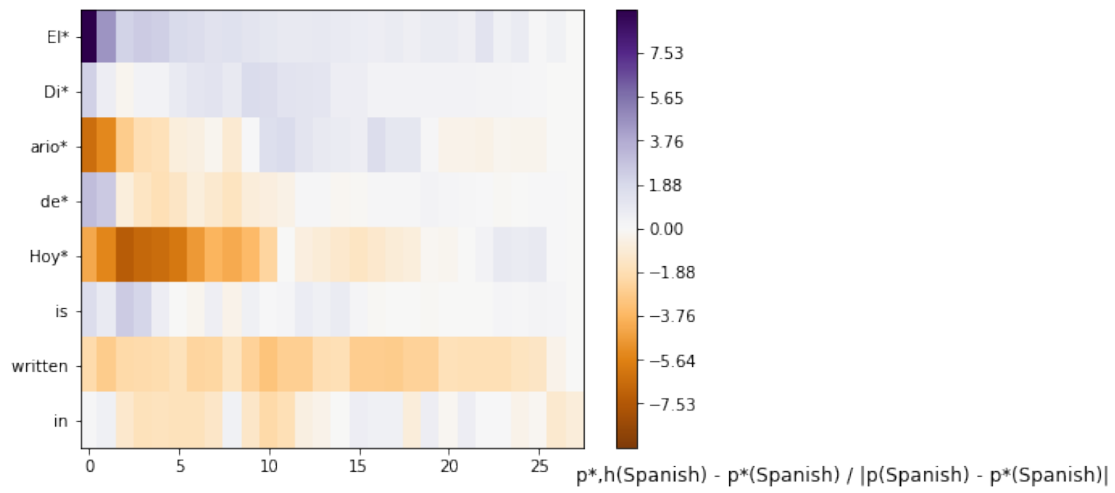


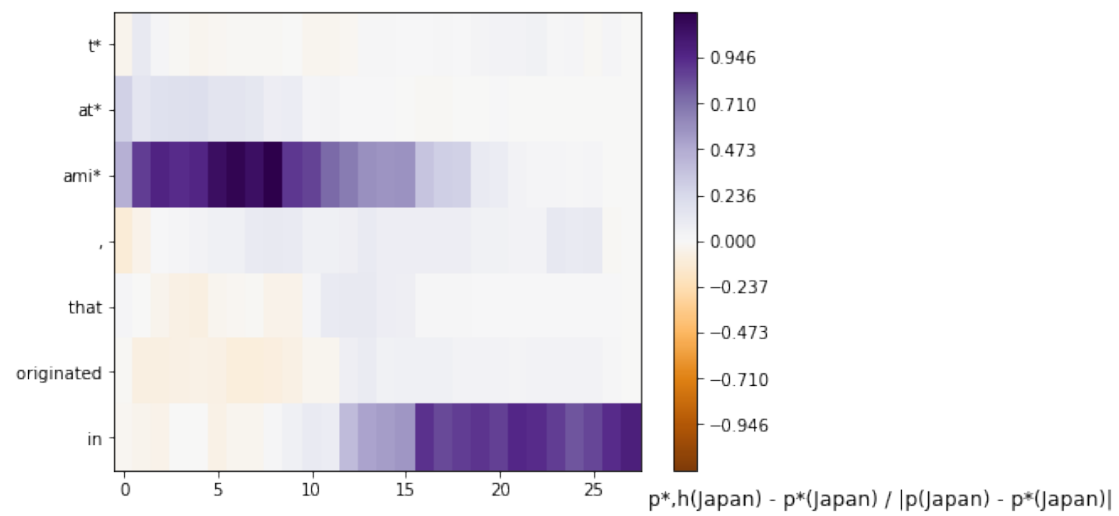
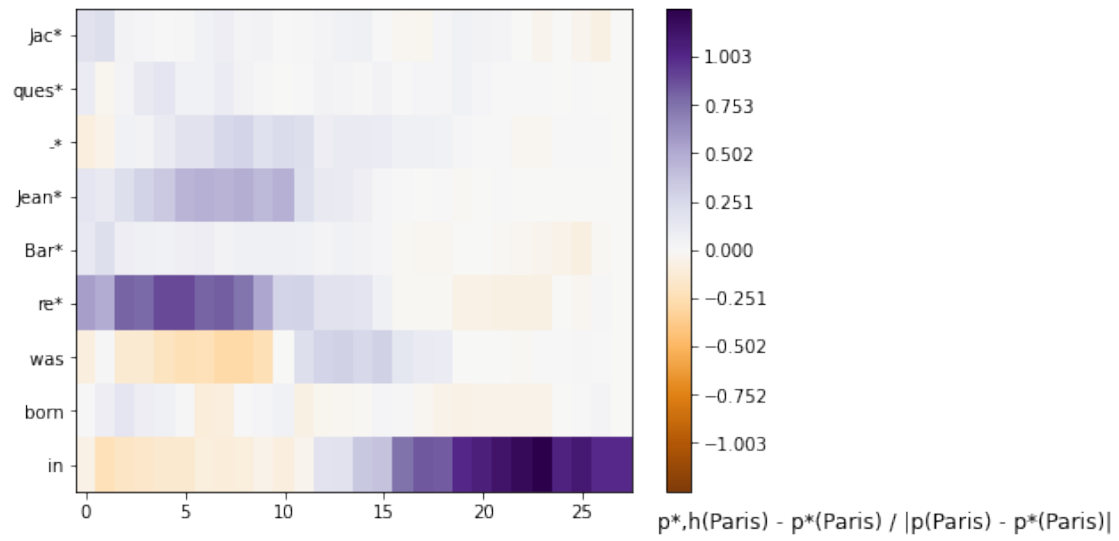


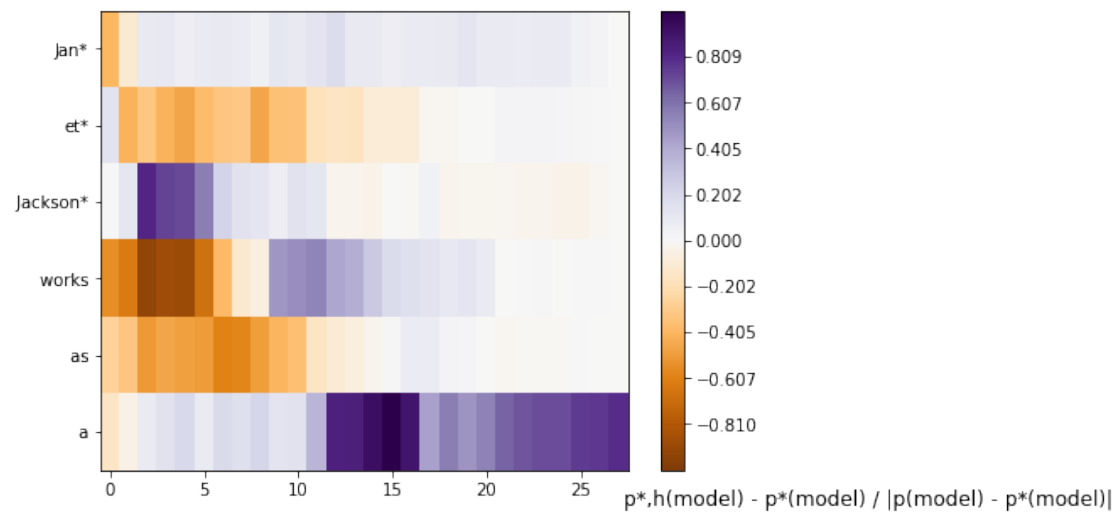
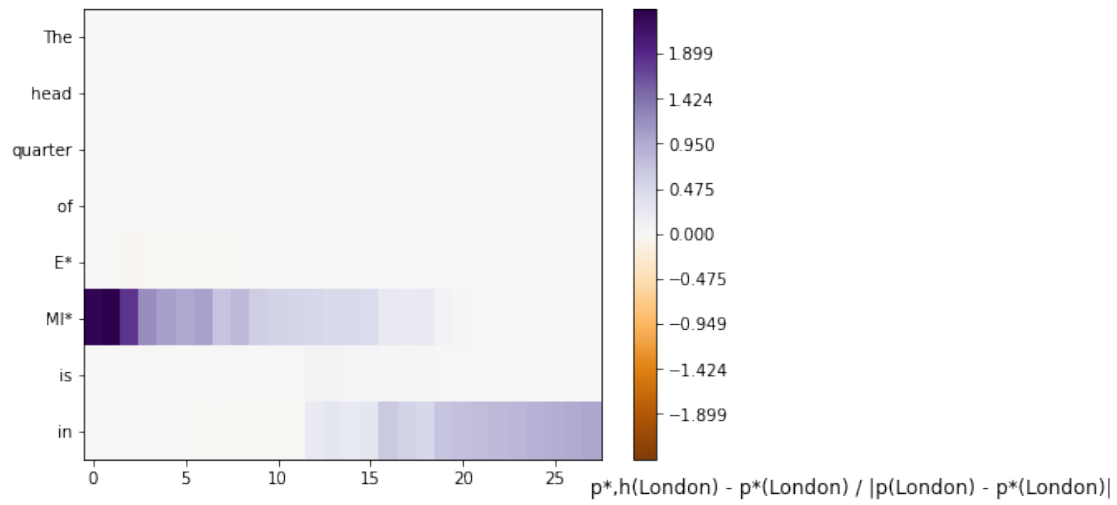


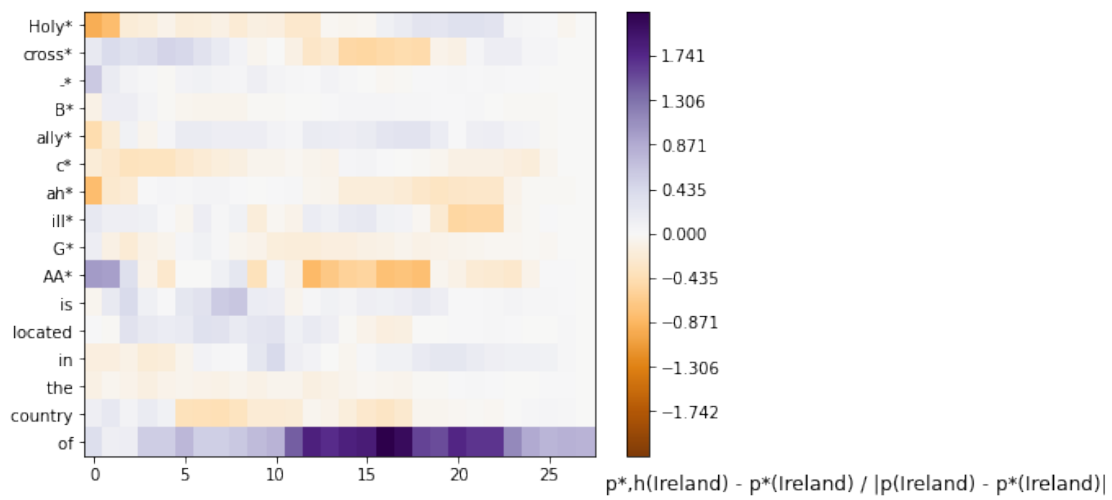
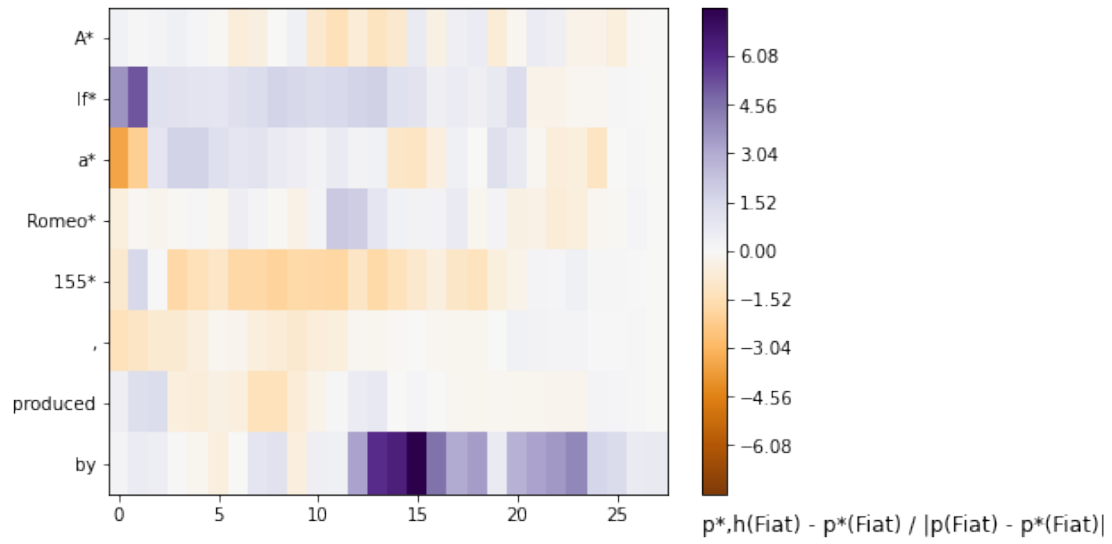


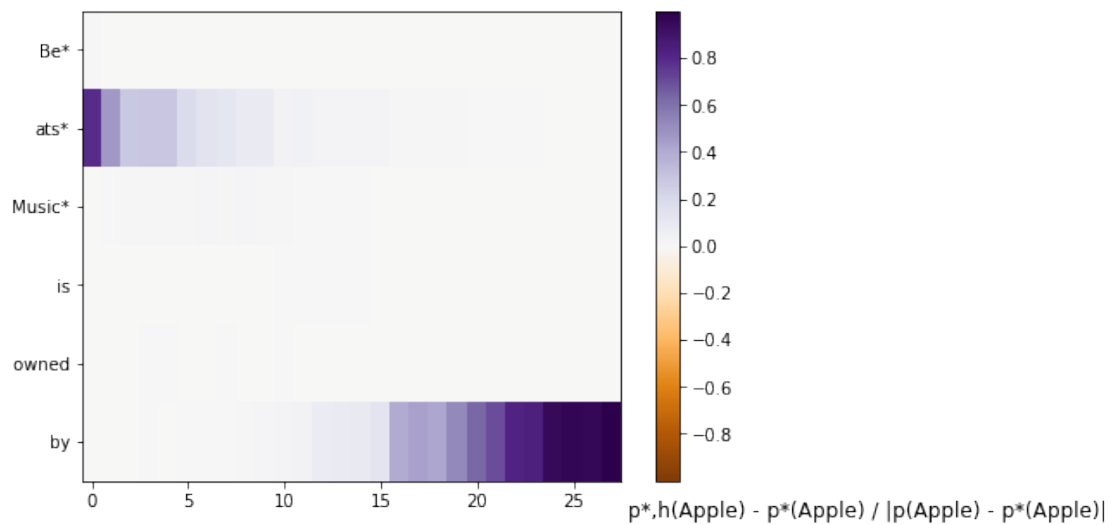
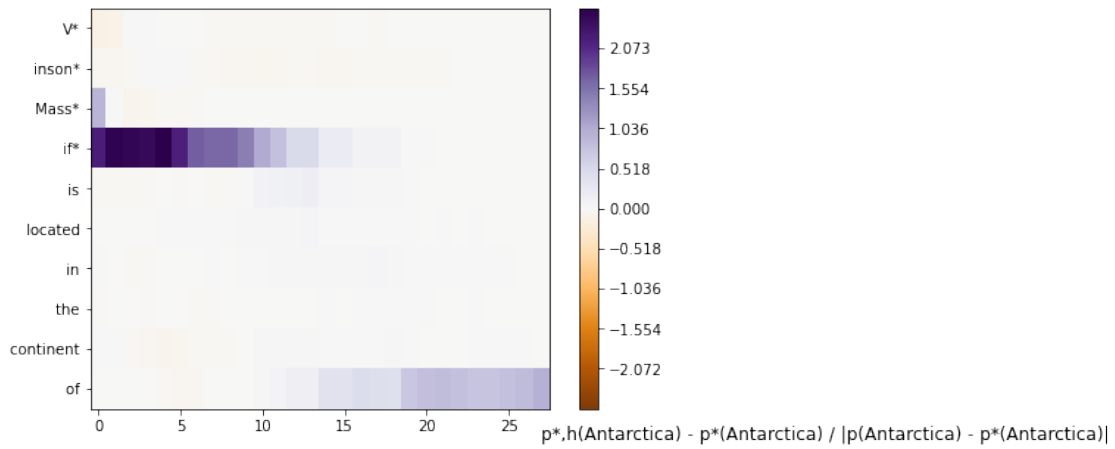
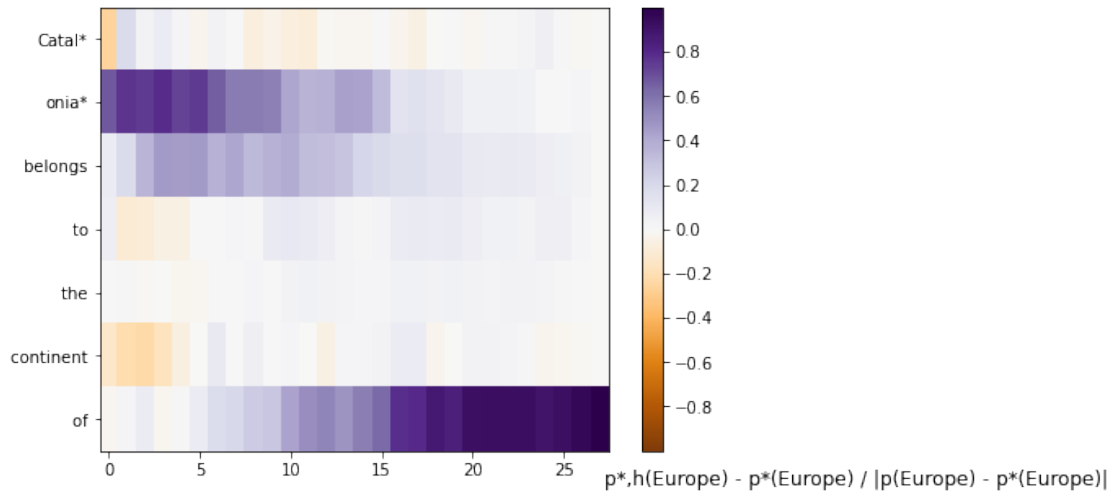


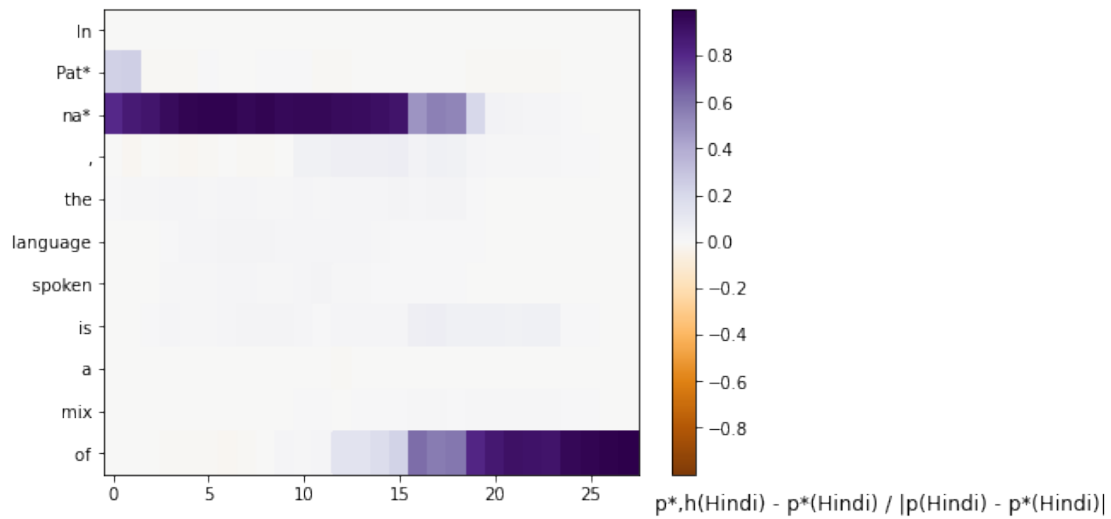
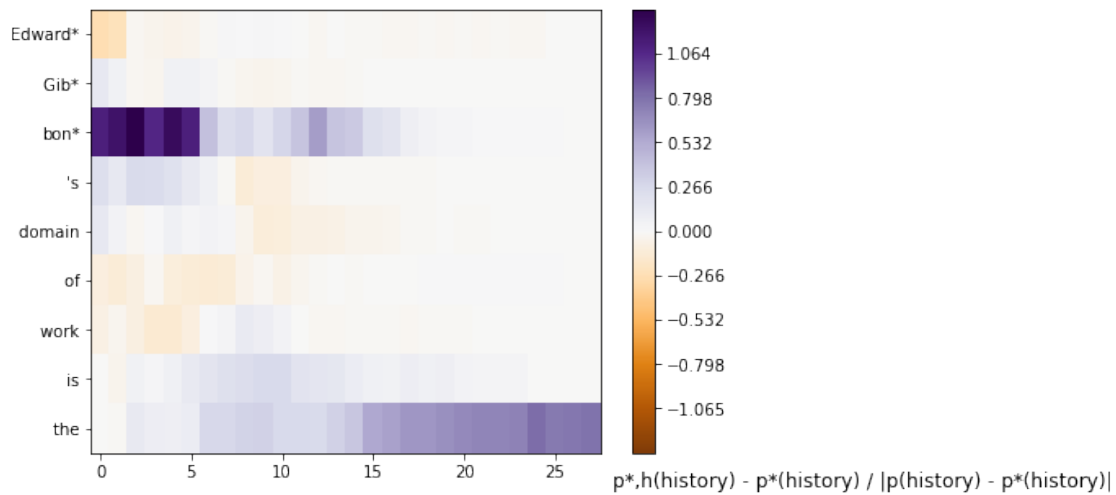












```
[9]: # zero cutoff
for prob, access in res:
    text, tokens, num_layers, _, _ = access
    x = num_layers
    y = tokens
    c = 0.001
    lower_bound = 0
    upper_bound = max(torch.max(prob).item(), c)
    incr = (upper_bound - lower_bound) / 10000
    token = text.split()[-1]
```