Mestrado em Bioinformática

Text Mining project pipeline

José Pedro Silva Freitas, PG42872

pg42872@alunos.uminho.pt

O presente documento tem engloba uma representação mais organizada da pipeline que foi realizada de modo a solucionar os problemas previstos para a realização deste projeto.
Como já referenciado no artigo, este projeto teve como principal objetivo a mineração de textos biomédicos de forma a encontrar referências acerca do metabolismo secundário de plantas, mais especificamente, na via metabólica relacionada com o composto resveratrol, de forma a facilitar o estudo deste tipo de metabolismo, assim como a sua interferência no fenótipo das plantas

In [11]:
```python
"""Imports"""

import tensorflow
import pandas
# import numpy as np
# from Bio.Entrez import efetch, read
from Bio import Entrez
import os
import sys
from web_pubmed_reader import get_data_from_pmid, \
    term_to_docs_all, pmids_to_docs
from dl_config import DLConfig
from wrappers.pandas_wrapper import docs_to_pandasdocs
# pandasframe_trimming,
# search_keyword_pandasframe
import string
from stop_words import get_stop_words
```

Como base para esta pipeline, foi usada uma ferramente já disponível, **BioTMPy**, e que foi desenvolvida com um propósito relativamente idêntico, a partir da qual se pode retirar algumas ilações.
Os módulos deste pacote incluem classes e funções que permitem a extração de documentos e artigos presentes em bases de dados, neste caso do PubMed, através do package **Biopython**, mais especificamente, o package **Bio.Entrez**.

In [17]:
```python
keywordlst = ["resveratrol",
              "resveratrol synthase",
              "stilbene synthase",
              "trihydroxystilbene",
              "p-coumaroyl-CoA",
              "viniferin",
              "piceatannol",
              "piceid",
              "pterostilbene",
              "pinostilbene",
              "chalcone resveratrol"]


def get_number_articles_per_term(term, email):
    Entrez.email = email
    Entrez.api_key = '0a925fca3ee9689bd778607f10e438fbfa09'
    search_results = Entrez.read(Entrez.esearch(db="pubmed", term=term))
    records = search_results["Count"]
    return int(records)


for keyword in keywordlst:
    print("Número de artigos encontrados para a keyword", keyword+":",
      get_number_articles_per_term(keyword, "pg42872@alunos.uminho.pt"))
```

```
Número de artigos encontrados para a keyword resveratrol: 14485
Número de artigos encontrados para a keyword resveratrol synthase: 52
Número de artigos encontrados para a keyword stilbene synthase: 287
Número de artigos encontrados para a keyword trihydroxystilbene: 542
Número de artigos encontrados para a keyword p-coumaroyl-CoA: 109
Número de artigos encontrados para a keyword viniferin: 319
Número de artigos encontrados para a keyword piceatannol: 839
Número de artigos encontrados para a keyword piceid: 721
Número de artigos encontrados para a keyword pterostilbene: 638
Número de artigos encontrados para a keyword pinostilbene: 35
Número de artigos encontrados para a keyword chalcone resveratrol: 68
```

Foi desenvolvida uma nova função - *get_all_data_from_term* - que suportasse não um número máximo de artigos definido pelo utilizador, mas sim o número máximo de artigos para um dado termo, isto sabendo que a função do **Bio.Entrez** tem um rate máximo de extração de cerca de 100000 ID's por cada iteração, e para prevenir os casos em que o número de artigos ultrapassa este threshold.

In [19]:
```python
def get_all_data_from_term(term, email):
    Entrez.email = email
    Entrez.api_key = '0a925fca3ee9689bd778607f10e438fbfa09'
    retmax = get_number_articles_per_term(term, email)
    records = []
    ub = 100000  # número máximo de artigos que a função esearch consegue capturar
    if retmax > ub:
        for startpos in range(0, retmax, ub):
            handle = Entrez.esearch(db="pubmed", retstart=startpos, retmax=ub, term=term, idtype="acc",
                                    sort="relevance")
            record = Entrez.read(handle)
            records.append(record)  # pode-se dar append ao record numa lista? SIM PODE-SE
            handle.close()
    else:
        handle = Entrez.esearch(db="pubmed", retmax=retmax, term=term, idtype="acc", sort='relevance')
        record = Entrez.read(handle)
        records.append(record)
        handle.close()

    size = 0
    result = []
    if len(records) > 1:  # para o caso do numero de artigos ter sido maior que 100000
        for record in records:  # irá ter mais que um record
            size += len(record["IdList"])
            result.extend(record["IdList"])  # será uma lista de listas de IDs
        flat_lst = []  # dar flat à lista de IDs
        for lst in result:
            for elem in lst:
                flat_lst.append(elem)
        return flat_lst  # [record["IdList"] for record in records]
    else:  # para o caso que foi menos de 10000, e por isso records tem apenas uma lista
        for record in records:  # so tem 1
            result.extend(record["IdList"])
            return result


for keyword in keywordlst:
    print("Lista de ID's encontrados para a keyword",keyword+":",
          get_all_data_from_term(keyword, "pg42872@alunos.uminho.pt")[:20])
```

```
Lista de ID's encontrados para a keyword resveratrol: ['33691210', '33550599', '34066865', '34064568', '3378177
0', '33229156', '33712142', '33675782', '33675348', '33631244', '33922475', '33471184', '33946456', '33921192',
'33388350', '33802331', '33103284', '33518308', '33669559', '33561994']
Lista de ID's encontrados para a keyword resveratrol synthase: ['26302213', '33444365', '25007571', '24427944',
'21717185', '21932029', '11804335', '32733034', '32664999', '9225853', '2450022', '26344106', '25764453', '2609
2168', '25873669', '25683151', '17571969', '29138972', '24196732', '33614263']
Lista de ID's encontrados para a keyword stilbene synthase: ['30611873', '31699028', '29589146', '31531782', '2
9372791', '24246671', '28168876', '27702992', '31925968', '32456002', '28332216', '28214727', '26048582', '2540
2946', '29529275', '30445464', '28181620', '25726587', '21843947', '26781778']
Lista de ID's encontrados para a keyword trihydroxystilbene: ['27882602', '33915131', '28672886', '26112149',
'27834152', '23042952', '22349988', '20715097', '21520490', '18927354', '19940103', '18324741', '17872969', '17
875610', '16199232', '21783793', '15809979', '15941348', '12776577', '9865343']
Lista de ID's encontrados para a keyword p-coumaroyl-CoA: ['28822035', '22168819', '24372757', '24654730', '268
26222', '18316744', '30814883', '25152451', '26400178', '26608135', '11038271', '33949064', '23510577', '336754
49', '23624856', '29794028', '20547380', '27956488', '27406087', '26527270']
Lista de ID's encontrados para a keyword viniferin: ['34144024', '33799534', '32394973', '32400050', '3211836
0', '33815533', '33773231', '31474855', '32502955', '33683879', '30400351', '29562698', '29556371', '29476302',
'28467376', '28782713', '32245220', '31547617', '27902942', '29155014']
Lista de ID's encontrados para a keyword piceatannol: ['32824997', '32886103', '32446839', '32524342', '3333501
5', '30940561', '31277394', '30551542', '30173053', '29764766', '32487622', '29792365', '30175773', '29057795',
'29041990', '28514198', '31696185', '29214257', '30178203', '31926274']
Lista de ID's encontrados para a keyword piceid: ['26013581', '24362562', '28980048', '31731792', '29989844',
'32389832', '27497981', '33667833', '33243482', '32835909', '32222939', '33229156', '32406041', '31755739', '33
092209', '31880518', '31494302', '30974157', '31085374', '33912552']
Lista de ID's encontrados para a keyword pterostilbene: ['33807403', '33739881', '31663469', '33171952', '33233
676', '32064876', '31935877', '31822918', '32125846', '31733141', '31452220', '31642668', '32832940', '3175433
3', '31078746', '32490497', '33830523', '31035507', '30578258', '33280250']
Lista de ID's encontrados para a keyword pinostilbene: ['33919396', '29268122', '26771130', '32919323', '326308
11', '30355929', '27756183', '24267561', '26990242', '31850235', '19443200', '25715927', '28242573', '3090954
4', '28279696', '30797367', '30471982', '25033820', '30021423', '33830523']
Lista de ID's encontrados para a keyword chalcone resveratrol: ['30058485', '26048788', '29327811', '21440448',
'32877711', '33453581', '31494471', '31817915', '30734608', '33130346', '27588016', '17655245', '28344479', '15
809979', '10426957', '25873669', '28981299', '23500714', '21717185', '23578977']
```

Com os PMID's de cada artigo do PubMed, o próximo objetivo será retirar o texto disponível nesses artigos, que serão os **títulos** e **abstracts** (resumos). A função *get_data_from_pmid* permite fazer exatamente esse processo, devolvendo um dicionário com duas keys e os dois values correspondentes na forma de string, procurando estes elementos em alguns dos difentes formatos que a função *entrez.efetch* retorna.

In [20]:
```python
resvid = get_all_data_from_term("resveratrol", "pg42872@alunos.uminho.pt")
print(resvid[0])
# Exemplo do output para o primeiro artigo encontrado referente ao resveratrol
get_data_from_pmid(resvid[0], "pg42872@alunos.uminho.pt")
```

```
33691210
```

Out[20]:
```
{'Title': 'Development of novel functional snacks containing nano-encapsulated resveratrol with anti-diabetic,
anti-obesity and antioxidant properties.',
 'Abstract': 'In this study, functional snacks with addition of nanoenapsuated resveratrol were prepared to eva
luate the nutraceutical and physical properties. The nanoencapsulated resveratrol was prepared from horse-chest
nut (HRP), water-chestnut (WRP) and lotus-stem starch particles (LRP) and added to the wheat flour at the level
of 0.4% for preparation of snacks by extrusing process. After extrusion, 43-53% and 5.42% of resveratrol was re
tained in snacks containing encapsulated and free resveratrol (FRP), respectively. The HRP, WRP and LRP showed
significantly less peak viscosities and less elastic behaviour than native product (NP) which can influence the
human sensory perception. The shift of few peaks towards higher wavelength and presence of additional peaks at
1384, 1229, 1513 and 1613\xa0cm<sup>-1</sup> depicts change in molecular pattern and presence of resveratrol in
functional snacks. The functional snacks containing encapsulated resveratrol showed significantly higher antiox
idant, anti-diabetic and anti-obesity properties than snacks containing no or free resveratrol.'}
```

Posto isto, o módulo Document com a classe document permite analisar e transformar as strings retiradas da função anterior, e a função *pmids_to_docs* devolve um objeto document por cada PMID fornecido.

Esta função necessita de um argumento adicional - **dl_config** - o qual representa uma classe criada à prori e que irá determinar o método de processamento dos documentos ao gosto do utilizador.

In [21]:
```python
dl = DLConfig()
# Exemplo do output para alguns dos artigos encontrado referente ao resveratrol
resvdoc = pmids_to_docs(resvid[0:20], "pg42872@alunos.uminho.pt", dl)
print(resvdoc)
print(len(resvdoc[0]))
```

```
([<data_structures.document.Document object at 0x000001E68CDDFEE0>, <data_structures.document.Document object a
t 0x000001E68EA03040>, <data_structures.document.Document object at 0x000001E68E9F72E0>, <data_structures.docum
ent.Document object at 0x000001E68E8AE0D0>, <data_structures.document.Document object at 0x000001E68EBBA7F0>, <
data_structures.document.Document object at 0x000001E68EB39820>, <data_structures.document.Document object at 0
x000001E68E9EC160>, <data_structures.document.Document object at 0x000001E68E835280>, <data_structures.documen
t.Document object at 0x000001E68ED53BE0>, <data_structures.document.Document object at 0x000001E68EE5DD00>, <da
ta_structures.document.Document object at 0x000001E68ED93910>, <data_structures.document.Document object at 0x0
0001E68E8A0EE0>, <data_structures.document.Document object at 0x000001E68ED413D0>, <data_structures.document.D
ocument object at 0x000001E68E9300D0>, <data_structures.document.Document object at 0x000001E68E92D400>, <data_
structures.document.Document object at 0x000001E68E847A00>, <data_structures.document.Document object at 0x0000
01E68ECF78B0>, <data_structures.document.Document object at 0x000001E68E0FBE0>, <data_structures.document.Docu
ment object at 0x000001E68E9C4130>, <data_structures.document.Document object at 0x000001E68F063250>], [])
20
```

Cada objeto Document pode ser analisado de acordo com o seu título, abstract, tokens, etc. A partir destes objetos será agora possível gerar um dataset com a estrutura desejadas, podendo conter apenas os objetos correspondentes a cada PMID (artigo) ou contendo já os textos de cada um.

O próximo objetivo passará ainda por obter todos os documentos já processados para todas as keyword sugeridas a cima........ Para facilitar o processo, e mais uma vez, como o objetivo será retirar **todos** os documentos disponíveis na base de dados, e para não haver a necessidade de passar o número total de documento aí existentes para as determinadas keywords, foi efetuada uma pequena alteração na função *term_to_docs*, para assim deixar de depender do argumento *ret_max* . Esta função passa assim a utilizar outra função criada anteriormente (*get_all_data_from_term()*) para retornar todos os PMIDs, e posteriormente gerar os objetos Document.

In [5]:
```python
resvdocs = term_to_docs_all("resveratrol", "pg42872@alunos.uminho.pt", dl)
resvsyntdocs = term_to_docs_all("resveratrol synthase", "pg42872@alunos.uminho.pt", dl)
stilbsyntdocs = term_to_docs_all("stilbene synthase", "pg42872@alunos.uminho.pt", dl)
trihydrostilbdocs = term_to_docs_all("trihydroxystilbene", "pg42872@alunos.uminho.pt", dl)
pcouCoAdocs = term_to_docs_all("p-coumaroyl-CoA", "pg42872@alunos.uminho.pt", dl)
vinidocs = term_to_docs_all("viniferin", "pg42872@alunos.uminho.pt", dl)
piceadocs = term_to_docs_all("piceatannol", "pg42872@alunos.uminho.pt", dl)
piddocs = term_to_docs_all("piceid", "pg42872@alunos.uminho.pt", dl)
pterodocs = term_to_docs_all("pterostilbene", "pg42872@alunos.uminho.pt", dl)
pinosdocs = term_to_docs_all("pinostilbene", "pg42872@alunos.uminho.pt", dl)
caldocs = term_to_docs_all("chalcone resveratrol", "pg42872@alunos.uminho.pt", dl)
```

Estando agora na posse dos objetos Document podemos passar para a criação dos primeiros datasets, nesta primeira instância, de um dataset para cada keyword.

Tendo em conta o objetivo deste trabalho e para facilitar vizualização, foi decidico montar um dataset onde fosse possível observar os Títulos e os Abstracts de cada documento, assim como o seu PMID e ainda a keyword através da qual foi encontrado.

O módulo **pandas** permite criar com facilidade **DataFrames** com diversos formatos e através de diferentes métodos. Uma nova função foi por essa razão criada com o nome de *docs_to_pandasdocs_idtitabs()* onde o utilizador deve passar uma lista de objetos Document previamente criada, assim como a keyword a partir da qual essa lista for criada, sob a forma de string.

Sendo assim foram criados os 11 datasets correspondentes às keywords utilizadas:

In [57]:
```python
def docs_to_pandasdocs_idtitabs(docs, keyword):
    doc_id = []
    titles = []
    abstracts = []
    for d in docs:  # são objeto Document
        doc_id.append(d.id)
        titles.append(d.title_string)  # lista de strings
        abstracts.append(d.abstract_string)  # lista de strings
    dicionario = {"Keyword": [keyword]*len(docs),
                  "Document": doc_id,
                  "Title": titles,
                  "Abstract": abstracts}  # será um dicionário onde os values serão uma lista de strings
    df = pandas.DataFrame(dicionario)
    df = pandas.DataFrame(data=df.values, columns=list(df.columns))
    # return titles, abstracts, doc_id
    return df


resv_df = docs_to_pandasdocs_idtitabs(resvdocs, "resveratrol")
resvsynt_df = docs_to_pandasdocs_idtitabs(resvsyntdocs, "resveratrol synthase")
stilbsynt_df = docs_to_pandasdocs_idtitabs(stilbsyntdocs, "stilbene synthase")
trihydrostilb_df = docs_to_pandasdocs_idtitabs(trihydrostilbdocs, "trihydroxystilbene")
pcouCoA_df = docs_to_pandasdocs_idtitabs(pcouCoAdocs, "p-coumaroyl-CoA")
vini_df = docs_to_pandasdocs_idtitabs(vinidocs, "viniferin")
picea_df = docs_to_pandasdocs_idtitabs(piceadocs, "piceatannol")
pid_df = docs_to_pandasdocs_idtitabs(piddocs, "piceid")
ptero_df = docs_to_pandasdocs_idtitabs(pterodocs, "pterostilbene")
pinos_df = docs_to_pandasdocs_idtitabs(pinosdocs, "pinostilbene")
cal_df = docs_to_pandasdocs_idtitabs(caldocs, "chalcone resveratrol")
```

O DataFrame global terá que ser criado através da concatenação dos diferentes datasets, *merge_pandasdocs()* é uma função simples e útil que usá a função *.concat()* do módulo **pandas** para unir **Dataframes** por intermédio das suas linhas, reiniciando também os seus índices.

Passando a lista das variáveis que seguram os objetos *DataFrame* criados a cima, obtemos o DataFrame global:

In [38]:
```python
def merge_pandasdocs(dataframes):
    res = pandas.concat(dataframes)
    res = res.reset_index(drop=True)
    return res


dflist = [resv_df, resvsynt_df, stilbsynt_df, pcouCoA_df, vini_df, picea_df,
          pid_df, ptero_df, pinos_df, cal_df]

big_df = merge_pandasdocs(dflist)
print(big_df)
print("O dataset completo final de todas as keywords terá uma dimensão de",big_df.shape)
```

```
                    Keyword  Document  \
0               resveratrol  33691210
1               resveratrol  33550599
2               resveratrol  34066865
3               resveratrol  34064568
4               resveratrol  33781770
...                     ...       ...
17548  chalcone resveratrol  12126798
17549  chalcone resveratrol  10563851
17550  chalcone resveratrol   9292395
17551  chalcone resveratrol   6427224
17552  chalcone resveratrol   2209605

                                                   Title  \
0      Development of novel functional snacks contain...
1      The synergistic antifungal activity of resvera...
2      Resveratrol Protects Cardiac Tissue in Experim...
3      Role of Resveratrol in Prevention and Control ...
4      Effect of resveratrol and mesenchymal stem cel...
...                                                  ...
17548  Prooxidant activity and cellular effects of th...
17549  PH-dependent forms of red wine anthocyanins as...
17550  Wine as a biological fluid : history , product...
17551  Purification and properties of a stilbene synt...
17552  The multifunctional 6-methylsalicylic acid syn...

                                                Abstract
0      In this study , functional snacks with additio...
1      Candida albicans is one of the most common cli...
2      Hypertension is one of the most prevalent and ...
3      Resveratrol is a phytoalexin produced by many ...
4      Osteoporosis is a skeletal disorder that is co...
...                                                  ...
17548  Dietary polyphenolics in fruits , vegetables ,...
17549  Anthocyanins are one of the main classes of fl...
17550  Wine has been part of human culture for 6,000 ...
17551  Stilbene synthase ( resveratrol -forming ) con...
17552  6-Methylsalicylic acid synthase ( MSAS ) from ...

[17553 rows x 4 columns]
O dataset completo final de todas as keywords terá uma dimensão de (17553, 4)
```

Analisando o recém criado Dataset, observa-se que muitos dos documentos se encontram repetidos, que pode ser explicado pelo facto de que o mesmo artigo pode retratar no seu corpo diferentes keywords, e portanto, os mesmos PMIDs são recolhidos em calls de keywords distintas, e que posteriormente irão criar linhas exatamente iguais no DataFrame.

Tendo isto em mente foi criada uma função *pandasframe_patterns()* , que retorna um novo DataFrame de acordo com a preferência do utilizador: se **unique** = **True** retorna um Dataframe apenas com as linhas que são unicas, ou seja, retira todos os duplicados, conservando apenas a primeira ocorrência dos mesmos; caso **unique** = **False**, faz exatamente o contrário e retorna as linhas que estavam duplicadas (apenas uma ocorrência),

In [12]:
```python
def pandasframe_patterns(dataframe, unique=True):
    """ se quiser só as rows que são unicas por default,
    unique = False, faz exatamente o contrário, retorna as rows que estavam duplicadas (apenas 1)"""
    if unique:
        df = dataframe.drop_duplicates(subset="Document")
    else:
        df = dataframe[dataframe.duplicated(subset="Document")]
        df = df.drop_duplicates()
    return df


duplicados = pandasframe_patterns(big_df, unique=False)
print("No dataset global existem", duplicados.shape[0], "artigos duplicados.")
```

```
No dataset global existem 1669 artigos duplicados.
```

O passo anterior é opcional neste estágio e pode (e deve) ser implementado mais à frente na pipeline.

O próximo passo deste projeto passa por reduzir a dimensão do dataset global, de forma a obter os documentos mais indicados possível de acordo com o tema em estudo, neste caso, a via de biosintese do **resveratrol** e vias associadas.
Assim, a construção de uma nova função foi mandatória, de forma a filtrar o grande DataFrame apenas para **documentos potencialmente relevantes**. Uma forma intuitiva de o fazer passará por apenas incluir artigos onde se encontrem, no seu Título ou Abstract, palavras de interesse que saibamos à priori que devem estar incluidas nesses textos para que sejam relevantes.
Para isso foi criada a função *pandasframe_trimming()*, que procura por um conjunto de palavras nos titlos e abstracts de cada linha do pandas DataFrame. Será então atribuido um score correspodente ao número de palavras que são encontradas no texto, expresso numa nova coluna "Occurrece", sendo no final as linhas do DataFrame ordenadas de acordo com esta.

In [39]:
```python
def pandasframe_trimming(dataframe, keywords):
    """Dada uma lista de keywords, procura cada uma delas em todos os titulos e abstracts presentes num dado da
    se numa dada linha, nenhuma das keywords se verificar, podemos eliminá-la"""
    counter = []
    for index, row in dataframe.iterrows():
        # print(index, row["Document"], type(row['Title']), type(row['Abstract']))
        number = 0
        for keyword in keywords:
            if keyword == row["Keyword"]:
                continue
            if keyword in str(row["Title"]).lower() or keyword in str(row["Abstract"]).lower():
                number += 1
        counter.append(number)
    df = dataframe.copy()
    df["Occurrence"] = counter
    df = df.sort_values(by="Occurrence", ascending=False)
    return df
```

Outro método passaria por procurar apenas um keyword específica nos textos do DataFrame, se a palavra constasse daí, seria atribuída a label **"Yes"**, caso contrário, a label **"No"**. No DataFrame final seriam apenas retornadas as linhas com a label igual a **"Yes"**.

In [15]:
```python
def search_keyword_pandasframe(dataframe, keyword):
    """dada uma keyword, procura essa keyword no titulo e abstract correspondente a todos os documentos do data
    dado e retorna os documentos onde essa keyword aparece, assim como o numero desses documentos """
    occ = []
    number = 0
    for index, row in dataframe.iterrows():
        # print(index, row["Document"], type(row['Title']), row['Abstract'])
        if row["Keyword"] != keyword:
            if keyword not in str(row["Title"]).lower() or keyword not in str(row["Abstract"]).lower():
                occ.append("No")
            else:
                number += 1
                occ.append("Yes")
        else:
            occ.append("Same")
    df = dataframe.copy()
    df["Occurrence"] = occ
    df = df.loc[df["Occurrence"] == "Yes"]
    df = df.drop(["Occurrence"], axis=1)
    return df, number


df_trimmed_onekey, num = search_keyword_pandasframe(big_df, "pathway")
print("Número de artigos com pathway:", num, "em cerca de", big_df.shape[0], "artigos")  # número de vezes (art
```

```
Número de artigos com pathway: 852 em cerca de 17553 artigos
```

A função a cima explicada e exemplificada pode ser interessante para o utilizador ter um ideia da qualidade do DataFrame global que foi criado anteriormente, em termos de uma keyword específica.

De seguida foi criada uma lista de keywords da forma mais completa possível, de forma a tentar abranger o maior número de artigos relevantes a este estudo, e assim ser um pouco mais restrito na sua procura.
Assim esta lista foi aplicada como argumento de *pandasframe_trimming*, sendo então retornado um novo **DaraFrame com o número de ocorrências destas palavras** nos seus textos.
Antes disso, foi também criado outro DataFrame através da mesma função, mas apenas com a lista de keywords *keywordlst*, que foi usada inicialmente para retirar os artigos do PubMed.

In [46]:
```python
findkeywordlst = ["resveratrol",
                  "trans-resveratrol",
                  "3,4',5-trihydroxystilbene",
                  "3,4',5-trihydroxy-trans-stilbene",
                  "resveratrol synthase",
                  "stilbene synthase",
                  "stilbenoids",
                  "stilbenes",
                  "trihydroxystilbene",
                  "resveratrol cleavage oxygenase",
                  "trihydroxystilbene synthase",
                  "resveratrol 3,5-O-dimethyltransferase",
                  "malonyl-CoA",
                  "p-coumaroyl-CoA",
                  "viniferin",
                  "piceatannol",
                  "piceid",
                  "pterostilbene",
                  "pinostilbene",
                  "phytoalexin",
                  "chalcone resveratrol",
                  "secondary metabolites",
                  "secondary metabolite biosynthesis",
                  "plants",
                  "phenotype"]

df_trimmed = pandasframe_trimming(big_df, keywordlst)
df_trimmed = df_trimmed.reset_index(drop=True)
df_trimmed_geral = pandasframe_trimming(big_df, findkeywordlst)
df_trimmed_geral = df_trimmed_geral.reset_index(drop=True)
```

A análise desde DataFrame permite verificar o grande número de artigos recolhidos que não correspondem em nenhuma medida ao assunto de interesse, tanto no DataFrame fruto da lista inicial como no DataFrame com as keywords mais gerais.
Isto pode ser observado através da grande maioria de artigos com menos de duas ocorrências de keywords de interesse, como se vai ver se seguida.

In [33]:
```python
# Para DataFrame com as keywords inicias
# retirar artigos duplicados de acordo com o PMID
df_trimmed = df_trimmed.drop_duplicates(subset="Document")

# Número de artigos com mais de uma ocorrência
df_trimmed_1 = df_trimmed[df_trimmed["Occurrence"] > 1]

# Número de artigos com mais de duas ocorrências
df_trimmed_2 = df_trimmed[df_trimmed["Occurrence"] > 2]

# Para DataFrame com as keywords específicas
# retirar artigos duplicados de acordo com o PMID
df_trimmed_geral = df_trimmed_geral.drop_duplicates(subset="Document")

# Número de artigos com mais de uma ocorrência
df_trimmed_geral_1 = df_trimmed_geral[df_trimmed_geral["Occurrence"] > 1]

# Número de artigos com mais de duas ocorrências
df_trimmed_geral_2 = df_trimmed_geral[df_trimmed_geral["Occurrence"] > 2]

# Número de artigos com mais de três ocorrências
df_trimmed_geral_3 = df_trimmed_geral[df_trimmed_geral["Occurrence"] > 3]

# Reiniciar indices das linhas
df_trimmed_geral_2 = df_trimmed_geral_2.reset_index(drop=True)
df_trimmed_geral_3 = df_trimmed_geral_3.reset_index(drop=True)
```

O DataFrame criado com as keywords inicias não foi satisfatório, pelo que será descartado.
Em relação às keywords mais específicas, foi filtrado para apenas os artigos com mais de 1, 2 ou 3 ocorrências: mais de 1 ocorrência resulta num número demasiado grande de artigos, enquanto que 3 num número demasiado baixo. Foi portanto utilizado o DataFrame com os artigos com mais de duas ocorrências de palavras.

Curação manual diretamente no ficheiro excel criado.
O objetivo será sempre encontrar artigos que sugiram, no seu abstract, que no seu corpo irão ser retratadas matérias relacionadas com o metabolismo do resveratrol.

In [30]:
```python
df = pandas.read_excel("C:/Users/Zé Freitas/Desktop/Mestrado/2°Semestre/Projeto/code/"
                       "datasets/dataset_geral_marta.xlsx",
                       usecols="C,J")
print(df.shpape)
```

Nesta primeria análise observou-se uma grande incidência de artigos não relevantes, o que seria de esperar, tendo em conta o próprio tema.
No entanto, achamos que estes valores poderiam ser melhorados, e assim ficou decidido adicionar novas palavras para uma filtração mais rigorosa do DataFrame global.

In [51]:
```python
newords = ["pathway",
           "biosynthesis",
           "synthesis"]  # 3 novas palavras

if newords[0] not in findkeywordlst:
    findkeywordlst.extend(newords)

# Atribuição de scores
df_trimmed_geral = pandasframe_trimming(big_df, findkeywordlst)
df_trimmed_geral = df_trimmed_geral.drop_duplicates(subset="Document")
df_trimmed_geral = df_trimmed_geral.reset_index(drop=True)

df_trimmed_geral_2 = df_trimmed_geral[df_trimmed_geral["Occurrence"] > 2]
df_trimmed_geral_3 = df_trimmed_geral[df_trimmed_geral["Occurrence"] > 3]
df_trimmed_geral_2 = df_trimmed_geral_2.reset_index(drop=True)
df_trimmed_geral_3 = df_trimmed_geral_3.reset_index(drop=True)
```

Para este caso ficou decidido avançar com o DataFrame filtrado para mais de 3 ocorrências.

Para facilitar a análise deste novo DataFrame, e prevendo que grande parte dos documentos presentes neste estarão já presentes no DataFrame anterior **df**, decidimos proceder ao merge dos dois, de modo a poder incluir nas linhas a label das relevâncias dos artigos já analisados, deixando em branco novos artigos.

In [58]:
```python
df_trimmed_geral_3["Document"] = df_trimmed_geral_3["Document"].astype(int)
df_merged = pandas.merge(df, df_trimmed_geral_3, on="Document", how="right")
# df_merged.to_excel("C:/Users/Zé Freitas/Desktop/Mestrado/2°Semestre/Projeto/"
#                    "code/datasets/dataset_geral_marta_v2.xlsx")
```

Os artigos com "Relevance" em branco (ou com "non" como atributo) neste novo DataFrame serão mais uma vez curados manualmente.

Ao efetuar o merge de ambos, houve alguns problemas de formatação do excel, por isso de seguida os DataFrames serão reorganizados.

In [59]:
```python
# Primeiro DataFrame
df_final_v1 = pandas.read_excel("C:/Users/Zé Freitas/Desktop/Mestrado/2°Semestre/"
                                "Projeto/code/datasets/dataset_geral_marta.xlsx",
                                usecols="B:E,J")
df_final_v1.rename(columns={'Unnamed: 9': 'Relevance'}, inplace=True)
df_relevant = df_final_v1[df_final_v1["Relevance"] == "relevant"]
df_relevant = df_relevant.reset_index(drop=True)
cols = list(df_final_v1.columns.values)
print(cols)

# Novo DataFrame com mais keywords
df_final_v2 = pandas.read_excel("C:/Users/Zé Freitas/Desktop/Mestrado/2°Semestre/Projeto/code/datasets/"
                                "dataset_geral_marta_v2.xlsx",
                                index_col=0, usecols="A:F")
df_final_v2.rename(columns={'Unnamed: 9': 'Relevance'}, inplace=True)
df_relevant_v2 = df_final_v2[df_final_v2["Relevance"] == "relevant"]
df_relevant_v2 = df_relevant_v2.reset_index(drop=True)
colsv2 = list(df_final_v2.columns.values)
print(colsv2)

# Organização do segundo DataFrame de acordo com o primeiro
df_final_v2 = df_final_v2[['Keyword', 'Document', 'Title', 'Abstract', 'Relevance']]
```

```
['Keyword', 'Document', 'Title', 'Abstract', 'Relevance']
['Document', 'Relevance', 'Keyword', 'Title', 'Abstract']
```

Como visto antes, estes dois DataFrames terão artigos repetidos, daí o merge ter resultado, portanto terá que se concatenar ambos para assim eliminar os duplicados e obtermos um DataFrame decente

In [60]:
```python
dfs_final = [df_final_v1, df_final_v2]

# Concatenar os dois DataFrames
df_finalissimo = merge_pandasdocs(dfs_final)

# Eliminação de duplicados de acordo com PMID
df_finalissimo = df_finalissimo.drop_duplicates(subset="Document")

# Reiniciar os indices das linhas
df_finalissimo = df_finalissimo.reset_index(drop=True)
```

Em jeito de finalização, pode ser revisto o número de documentos que foram classificados como relevantes no nosso **DataSet Final**.

In [61]:
```python
df_relevant_finalissimo = df_finalissimo[df_finalissimo["Relevance"] == "relevant"]
print(df_relevant_finalissimo.shape[0])
print("Número de artigos relevantes:", df_relevant_finalissimo.shape[0], "em cerca de",
      df_finalissimo.shape[0], "artigos")
```

```
193
Número de artigos relevantes: 193 em cerca de 606 artigos
```

Deep Learning

Hierarchical Attention Networks
w/ parametrs optmized

Com stops words gerais.

In [12]:
```python
"""Imports"""

import tensorflow
import numpy as np
import random
import os

seed_value = 11111
np.random.seed(seed_value)
random.seed(seed_value)
tensorflow.random.set_seed(seed_value)

from dl import DL_preprocessing
from dl_models import HAN_opt
from embeddings import compute_embedding_matrix
from sklearn.metrics import accuracy_score
from sklearn.metrics import precision_score
from sklearn.metrics import recall_score
from sklearn.metrics import f1_score
from sklearn.metrics import confusion_matrix
from tensorflow.keras.callbacks import EarlyStopping, ModelCheckpoint
from dl import plot_training_history
from dl_config import DLConfig
from dl import average_precision  # vai dar uma prob para um dado doc ser relevant ou nao
from tensorflow.keras.preprocessing import text
from dl import plot_roc_n_pr_curves
from sklearn.model_selection import train_test_split
# noinspection PyUnresolvedReferences
from tensorflow.compat.v1 import ConfigProto
# noinspection PyUnresolvedReferences
from tensorflow.compat.v1 import InteractiveSession
import tensorflow.compat.v1.keras.backend as K
import wget
from nltk.corpus import stopwords

stops = set(stopwords.words("english"))

config = ConfigProto()
config.gpu_options.allow_growth = True
session = InteractiveSession(config=config)
K.set_session(session)

print(tensorflow.__version__)

print("Num GPUs Available: ", len(tensorflow.config.experimental.list_physical_devices('GPU')))

multiple_gpus = [0]
if multiple_gpus:
    devices = []
    for gpu in multiple_gpus:
        devices.append('/gpu:' + str(gpu))
    strategy = tensorflow.distribute.MirroredStrategy(devices=devices)
else:
    # Get the GPU device name.
    device_name = tensorflow.test.gpu_device_name()
    # The device name should look like the following:
    if device_name == '/device:GPU:0':
        print('Using GPU: {}'.format(device_name))
    else:
        raise SystemError('GPU device not found')

    os.environ["CUDA_VISIBLE_DEVICES"] = device_name
    os.environ["CUDA_DEVICE_ORDER"] = "PCI_BUS_ID"
```

```
2.4.1
Num GPUs Available:  0
WARNING:tensorflow:Some requested devices in `tf.distribute.Strategy` are not visible to TensorFlow: /job:local
host/replica:0/task:0/device:GPU:0
INFO:tensorflow:Using MirroredStrategy with devices ('/job:localhost/replica:0/task:0/device:GPU:0',)
C:\Users\Zé Freitas\AppData\Roaming\Python\Python38\site-packages\tensorflow\python\client\session.py:1761: Use
rWarning: An interactive session is already active. This can cause out-of-memory errors in some cases. You must
explicitly call `InteractiveSession.close()` to release resources held by the other session(s).
  warnings.warn('An interactive session is already active. This can '
```

In [63]:
```python
dl_config = DLConfig(remove_punctuation="HTML", stop_words=stops, lower=True, split_by_hyphen=True)


def pandas_column_aslist(dataframe, column):
    lst = dataframe[column].tolist()
    return lst  # lista de strings


df = pandas.read_excel("C:/Users/Zé Freitas/Desktop/Mestrado/2°Semestre/Projeto/code/datasets/dataset_final.xls
                       index_col=0)
idsfinal = pandas_column_aslist(df, "Document")
docsfinal = pmids_to_docs(idsfinal, "pg42872@alunos.uminho.pt", dl_config)
docsfinal = docsfinal[0]
print(docsfinal[0].abstract_string)
dataset_docs = docs_to_pandasdocs(docsfinal)
# dataset_docs.to_csv("C:/Users/Zé Freitas/Desktop/Mestrado/2°Semestre/Projeto/code/datasets/dataset_teste_semp
```

```
stilbenes , important secondary metabolites grapevine , represent central phytoalexins therefore constitute imp
ortant element basal immunity . study , potential genetic variation vitis vinifera ssp . sylvestris , ancestor
cultivated grapevine , sought respect output stilbenes potential use resistance breeding . considerable variati
on stilbene inducibility identified v. vinifera ssp . sylvestris . genotypic differences abundance profiles sti
lbenes induced response uv - c pulse shown . two clusters stilbene 'chemovars ' emerged : one cluster showed qu
ick strong accumulation stilbenes , almost exclusively form non - glycosylated resveratrol viniferin , second c
luster accumulated fewer stilbenes relatively high proportions piceatannol glycosylated piceid . 86 genotypes ,
time dependence stilbene pattern observed : piceid , resveratrol , piceatannol accumulated earlier , whereas vi
niferins found later . observed genotypic differences stilbene accumulation preceded differential accumulation
transcripts chalcone synthase ( chs ) stilbene - related genes : phenylalanine ammonium lyase ( pal ) , stilben
e synthase ( stsy ) , resveratrol synthase ( rs ) . screen population respect susceptibility downy mildew grape
vine ( plasmopara viticola ) revealed considerable variability . subpopulation genotypes high stilbene inducibi
lity significantly less susceptible compared low - stilbene genotypes , representative genotypes could shown in
ducibility stilbene synthase uv correlated inducibility pathogen .
```

In [64]:
```python
x_total = dataset_docs
y_total = pandas.read_excel("C:/Users/Zé Freitas/Desktop/Mestrado/2°Semestre/Projeto/code/datasets/dataset_fina
                            index_col=1, usecols="B:F")

print(x_total)


def relevances_to_pandas(dataframe, relevances):
    indexes = list(dataframe.index.values)
    labels = []
    for r in relevances:
        if r == "not relevant":
            labels.append(0)
        elif r == "relevant":
            labels.append(1)
    return pandas.Series(labels, index=indexes, name='Label')


relevance = pandas_column_aslist(y_total, "Relevance")
y_total = relevances_to_pandas(y_total, relevance)
print(y_total)

print(x_total["Document"][0].title_string)
print(x_total["Document"][0].abstract_string)
```

```
                                                   Document
25873669  <data_structures.document.Document object at 0...
24269870  <data_structures.document.Document object at 0...
17655245  <data_structures.document.Document object at 0...
16663649  <data_structures.document.Document object at 0...
33907682  <data_structures.document.Document object at 0...
...                                                     ...
32079399  <data_structures.document.Document object at 0...
26976595  <data_structures.document.Document object at 0...
20138774  <data_structures.document.Document object at 0...
27936616  <data_structures.document.Document object at 0...
28009095  <data_structures.document.Document object at 0...

[606 rows x 1 columns]
25873669    0
24269870    0
17655245    1
16663649    1
33907682    0
           ..
32079399    1
26976595    0
20138774    1
27936616    1
28009095    1
Name: Label, Length: 606, dtype: int64
genetic diversity stilbene metabolism vitis sylvestris .
stilbenes , important secondary metabolites grapevine , represent central phytoalexins therefore constitute imp
ortant element basal immunity . study , potential genetic variation vitis vinifera ssp . sylvestris , ancestor
cultivated grapevine , sought respect output stilbenes potential use resistance breeding . considerable variati
on stilbene inducibility identified v. vinifera ssp . sylvestris . genotypic differences abundance profiles sti
lbenes induced response uv - c pulse shown . two clusters stilbene 'chemovars ' emerged : one cluster showed qu
ick strong accumulation stilbenes , almost exclusively form non - glycosylated resveratrol viniferin , second c
luster accumulated fewer stilbenes relatively high proportions piceatannol glycosylated piceid . 86 genotypes ,
time dependence stilbene pattern observed : piceid , resveratrol , piceatannol accumulated earlier , whereas vi
niferins found later . observed genotypic differences stilbene accumulation preceded differential accumulation
transcripts chalcone synthase ( chs ) stilbene - related genes : phenylalanine ammonium lyase ( pal ) , stilben
e synthase ( stsy ) , resveratrol synthase ( rs ) . screen population respect susceptibility downy mildew grape
vine ( plasmopara viticola ) revealed considerable variability . subpopulation genotypes high stilbene inducibi
lity significantly less susceptible compared low - stilbene genotypes , representative genotypes could shown in
ducibility stilbene synthase uv correlated inducibility pathogen .
```

In [65]:
```python
X_train_df, X_test_df, y_train_df, y_test_df = train_test_split(x_total, y_total, test_size=0.3,
                                                                random_state=42, stratify=y_total)
print(X_train_df.shape)
print(X_test_df.shape)
print(y_train_df.shape)
print(y_test_df.shape)
```

```
(424, 1)
(182, 1)
(424,)
(182,)
```

In [66]:
```python
model_name = "HC_pro"
dl_config = DLConfig(model_name=model_name, seed_value=seed_value)
dl_config.stop_words = stops
dl_config.lower = True
dl_config.remove_punctuation = False
dl_config.split_by_hyphen = True
dl_config.lemmatization = False
dl_config.stems = False

#Parameters
dl_config.padding = 'post'            #'pre' -> default; 'post' -> alternative
dl_config.truncating = 'post'         #'pre' -> default; 'post' -> alternative      #####
dl_config.oov_token = 'OOV'

dl_config.epochs = 50
dl_config.batch_size = 32     # e aumentar o batch
dl_config.learning_rate = 0.001   #experimentar diminuir

dl_config.max_sent_len = 50      #sentences will have a maximum of "max_sent_len" words    #400/500
dl_config.max_nb_words = 100_000      #it will only be considered the top "max_nb_words" words in the dataset
dl_config.max_nb_sentences = 15    # set only for the hierarchical attention model!!!

dl_config.embeddings = 'biowordvec'

if not os.path.isdir('./embeddings'):
    os.mkdir('./embeddings')

# if dl_config.embeddings == 'biowordvec':   #200 dimensions
#     if not os.path.isfile('./embeddings/biowordvec'):
#         url = "https://ndownloader.figshare.com/files/12551780"
#         output_directory = "./embeddings/biowordvec"
#         wget.download(url, out=output_directory)
    #    !wget -O ./embeddings/biowordvec https://ndownloader.figshare.com/files/12551780
dl_config.embedding_path = './embeddings/biowordvec'
dl_config.embedding_dim = 200
dl_config.embedding_format = 'word2vec'

dl_config.keras_callbacks = True

if dl_config.keras_callbacks:
    dl_config.patience = 5   #early-stopping patience
    checkpoint_path = str(dl_config.model_id_path) + '\checkpoint.hdf5'
    keras_callbacks = [
            EarlyStopping(monitor='val_loss', mode='min', verbose=1, patience=dl_config.patience),
           ModelCheckpoint(checkpoint_path, monitor='val_loss', mode='min', verbose=1, save_best_only=True)
    ]
else:
    keras_callbacks = None
```

In [67]:
```python
dl_config.tokenizer = text.Tokenizer(num_words=dl_config.max_nb_words, oov_token=dl_config.oov_token)

dl_config.validation_percentage = 30

X_train, y_train, x_val, y_val = DL_preprocessing(X_train_df, y_train_df,
                                                  dl_config, dataset='train',
                                                  validation_percentage = dl_config.validation_percentage,
                                                  seed_value=dl_config.seed_value)
print(X_train[0][0])
```

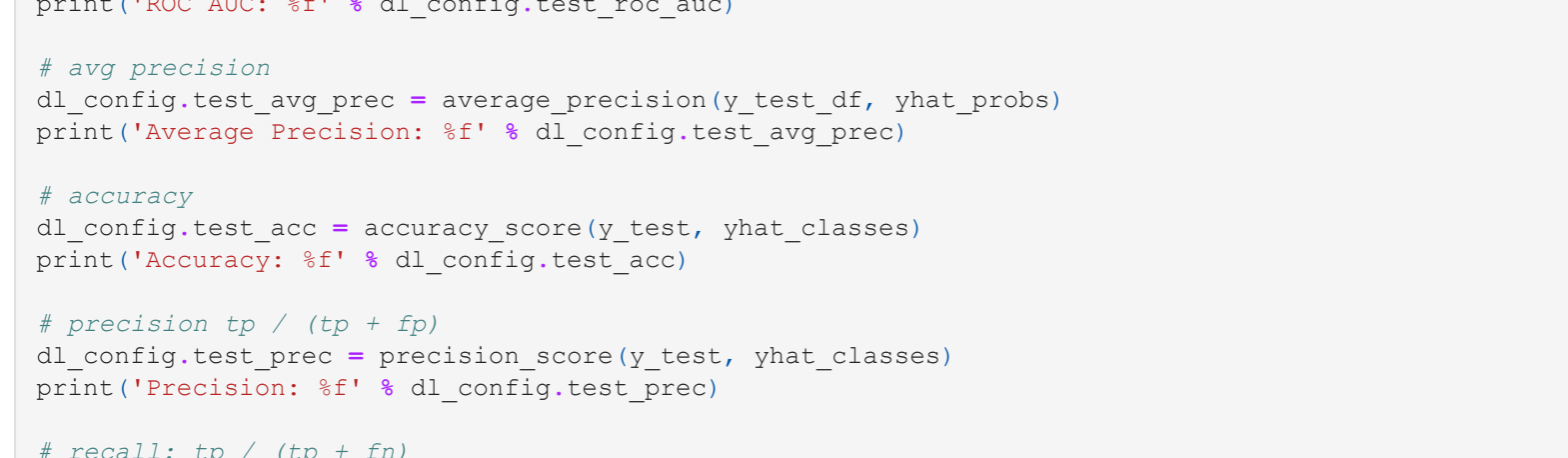
```
Found 7907 unique tokens.
Index of Unknown Words: 1
Training set with 297 samples
Validation set with 127 samples
[  22    5   20  698 1122 3434  178   20  698  431  497  333 4734  164
   86    1    0    0    0    0    0    0    0    0    0    0    0    0
    0    0    0    0    0    0    0    0    0    0    0    0    0    0
    0    0    0    0    0    0    0    0]
```

In [23]:
```python
dl_config.embedding_matrix = compute_embedding_matrix(dl_config, embeddings_format = dl_config.embedding_format
# quantas palavras encontrou no vocabulario do embedding

print(dl_config.embedding_matrix.shape)
print(dl_config.embedding_matrix)
```

```
Creating Embedding Matrix...
Embedding Matrix Created
------------------------
number of null word embeddings: 350 in a total of 7907 words (4.43%)
words not found: 349
e.g. 10 words not found in the index :  ['hydroxyferuloyl' "'superior" "'jingzaojing" 'dk18' 'pholidotol'
 'dhs2001' '882000m' 'furtherstudy' 'metabolitism' 'piceides' '6β'
 'danfeng' 'myba2' 'sts36' 'spirotropis']
(7908, 200)
[[ 0.          0.          0.         ...  0.          0.
   0.        ]
 [ 0.          0.          0.         ...  0.          0.
   0.        ]
 [-0.0848686  -0.0417668  -0.3709338  ...  0.07843895  0.08479623
   0.54163074]
 ...
 [ 0.15182443 -0.13880545  0.16630234 ... -0.29269406 -0.08847184
  -0.05204243]
 [-0.1121414  -0.08872321 -0.33860227 ... -0.34313729  0.25840917
  -0.20746794]
 [ 0.26528463  0.19522475 -0.05939044 ...  0.18617576 -0.05572284
  -0.12646711]]
```

In [24]:
```python
if multiple_gpus:
    with strategy.scope():
        model = HAN_opt(dl_config.embedding_matrix, dl_config, learning_rate=dl_config.learning_rate,
                                               seed_value=dl_config.seed_value)
else:
    model = HAN_opt(dl_config.embedding_matrix, dl_config, learning_rate=dl_config.learning_rate,
                                               seed_value=dl_config.seed_value)


history = model.fit(X_train, y_train,
                    epochs=dl_config.epochs,
                    batch_size=dl_config.batch_size,
                    validation_data=(x_val,y_val),
                    callbacks=keras_callbacks)

if dl_config.keras_callbacks:
    model.load_weights(checkpoint_path)

train_loss, dl_config.train_acc = model.evaluate(X_train, y_train, verbose=0, batch_size = dl_config.batch_size

print('Training Loss: %.3f' % (train_loss))
print('Training Accuracy: %.3f' % (dl_config.train_acc))

plot_training_history(history_dict=history, dl_config=dl_config)
```

```
Model: "model_3"
_________________________________________________________________
Layer (type)                 Output Shape              Param #
=================================================================
input_4 (InputLayer)         [(None, 15, 50)]          0
_________________________________________________________________
time_distributed_1 (TimeDist (None, 15, 256)           1984544
_________________________________________________________________
spatial_dropout1d_3 (Spatial (None, 15, 256)           0
_________________________________________________________________
bidirectional_3 (Bidirection (None, 15, 512)           1050624
_________________________________________________________________
attention_with_context_3 (At (None, 512)               263168
_________________________________________________________________
dense_1 (Dense)              (None, 1)                 513
=================================================================
Total params: 3,298,849
Trainable params: 1,717,249
Non-trainable params: 1,581,600
_________________________________________________________________
None
Epoch 1/50
10/10 [==============================] - 59s 2s/step - loss: 0.6597 - accuracy: 0.6348 - val_loss: 0.6267 - val
_accuracy: 0.708
7□□□□□□□□□□□□□□□□□□□□


Epoch 00001: val_loss improved from inf to 0.62668, saving model to models\HC_pro\HC_pro_0\checkpoint.hdf5
Epoch 2/50
10/10 [==============================] - 19s 2s/step - loss: 0.6003 - accuracy: 0.6846 - val_loss: 0.5857 - val
_accuracy: 0.708
7□□□□□□□□□□□□□□□□□□□□


Epoch 00002: val_loss improved from 0.62668 to 0.58568, saving model to models\HC_pro\HC_pro_0\checkpoint.hdf5
Epoch 3/50
10/10 [==============================] - 15s 2s/step - loss: 0.5011 - accuracy: 0.7421 - val_loss: 0.6630 - val
_accuracy: 0.724
4□□□□□□□□□□□□□□□□□□□□


Epoch 00003: val_loss did not improve from 0.58568
Epoch 4/50
10/10 [==============================] - 14s 1s/step - loss: 0.3875 - accuracy: 0.8029 - val_loss: 0.5964 - val
_accuracy: 0.755
9□□□□□□□□□□□□□□□□□□□□


Epoch 00004: val_loss did not improve from 0.58568
Epoch 5/50
10/10 [==============================] - 15s 1s/step - loss: 0.4476 - accuracy: 0.7862 - val_loss: 0.6795 - val
_accuracy: 0.716
5□□□□□□□□□□□□□□□□□□□□


Epoch 00005: val_loss did not improve from 0.58568
Epoch 6/50
10/10 [==============================] - 15s 1s/step - loss: 0.4363 - accuracy: 0.8215 - val_loss: 0.9600 - val
_accuracy: 0.732
3□□□□□□□□□□□□□□□□□□□□


Epoch 00006: val_loss did not improve from 0.58568
Epoch 7/50
10/10 [==============================] - 16s 2s/step - loss: 0.4547 - accuracy: 0.7943 - val_loss: 0.6681 - val
_accuracy: 0.740
2□□□□□□□□□□□□□□□□□□□□


Epoch 00007: val_loss did not improve from 0.58568
Epoch 00007: early stopping
```

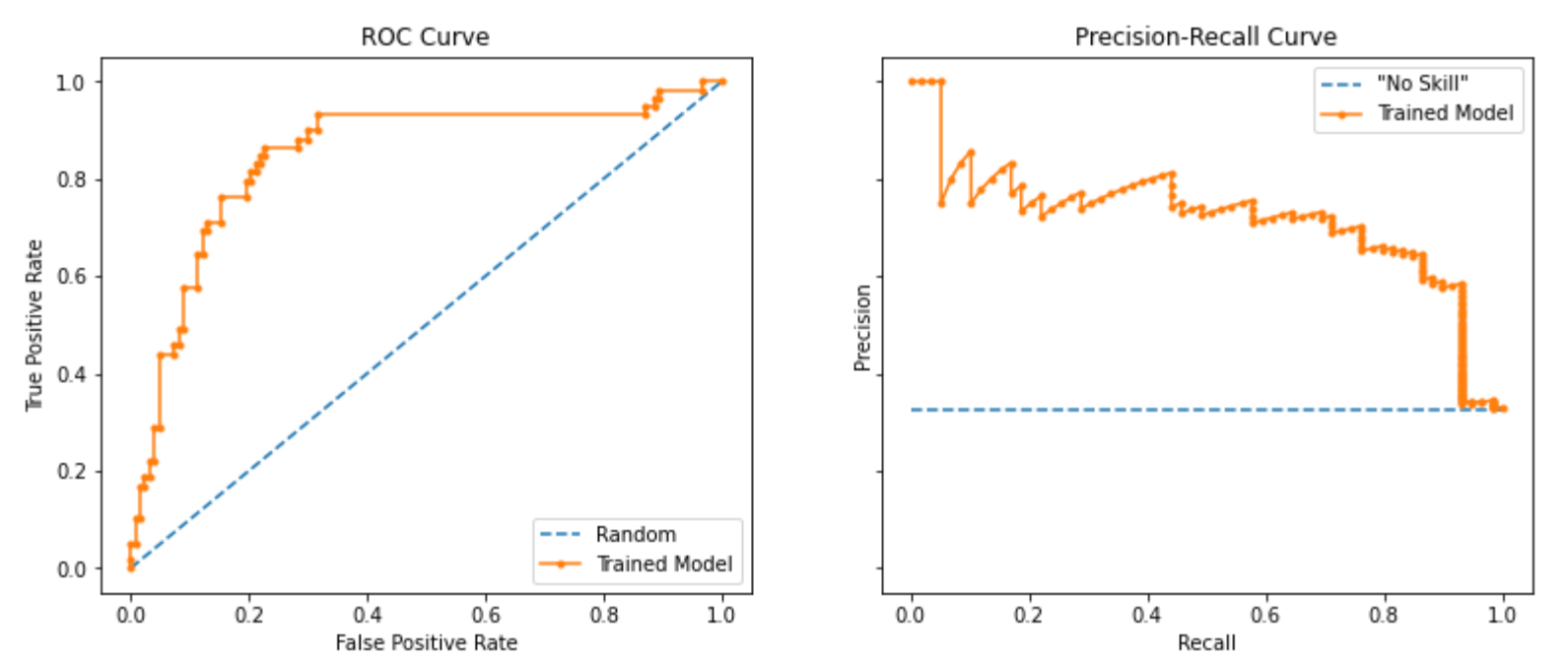


In [25]:
```python
x_test, y_test = DL_preprocessing(X_test_df, y_test_df, dl_config, dataset='test')

yhat_probs = model.predict(x_test, verbose=0)
yhat_probs = yhat_probs[:, 0]

yhat_classes = np.where(yhat_probs > 0.5, 1, yhat_probs)
yhat_classes = np.where(yhat_classes < 0.5, 0, yhat_classes).astype(np.int64)
```

```
Index of Unknown Words: 1
```

In [26]:
```python
dl_config.test_roc_auc, dl_config.test_pr_auc = plot_roc_n_pr_curves(y_test, yhat_probs,dl_config = dl_config)
# ROC AUC
print('ROC AUC: %f' % dl_config.test_roc_auc)

# avg precision
dl_config.test_avg_prec = average_precision(y_test_df, yhat_probs)
print('Average Precision: %f' % dl_config.test_avg_prec)

# accuracy
dl_config.test_acc = accuracy_score(y_test, yhat_classes)
print('Accuracy: %f' % dl_config.test_acc)

# precision tp / (tp + fp)
dl_config.test_prec = precision_score(y_test, yhat_classes)
print('Precision: %f' % dl_config.test_prec)

# recall: tp / (tp + fn)
dl_config.test_recall = recall_score(y_test, yhat_classes)
print('Recall: %f' % dl_config.test_recall)

# f1: 2 tp / (2 tp + fp + fn)
dl_config.test_f1_score = f1_score(y_test, yhat_classes)
print('F1 score: %f' % dl_config.test_f1_score)

# confusion matrix
matrix = confusion_matrix(y_test, yhat_classes)
matrix = np.flip(matrix)
print('Confusion Matrix:\n %s \n' % matrix)
```

```
ROC AUC: 0.848009
Average Precision: 0.721673
Accuracy: 0.719780
Precision: 0.722222
Recall: 0.220339
F1 score: 0.337662
Confusion Matrix:
 [[118   5]
 [ 46  13]]
```



Hierarchical Attention LSTM

In [27]:
```python
"""Imports"""

from dl_models import Hierarchical_Attention_LSTM
```

In [28]:
```python
from nltk.corpus import stopwords
stops = set(stopwords.words("english"))

dl_config = DLConfig(remove_punctuation="HTML", stop_words=stops, lower=True, split_by_hyphen=True)

df = pandas.read_excel("C:/Users/Zé Freitas/Desktop/Mestrado/2°Semestre/Projeto/code/datasets/dataset_final.xls
                    index_col=0)
idsfinal = pandas_column_aslist(df, "Document")
docsfinal = pmids_to_docs(idsfinal, "pg42872@alunos.uminho.pt", dl_config)
docsfinal = docsfinal[0]
docsfinal[0].abstract_string
dataset_docs = docs_to_pandasdocs(docsfinal)

x_total = dataset_docs
print(x_total)

y_total = pandas.read_excel("C:/Users/Zé Freitas/Desktop/Mestrado/2°Semestre/Projeto/code/datasets/dataset_fina
                         index_col=1, usecols="B:F")

print(x_total)

relevance = pandas_column_aslist(y_total, "Relevance")
y_total = relevances_to_pandas(y_total, relevance)
print(y_total)

print(x_total["Document"][0].title_string)
print(x_total["Document"][0].abstract_string)

X_train_df, X_test_df, y_train_df, y_test_df = train_test_split(x_total, y_total, test_size=0.3, random_state=
                                                              stratify=y_total)
print(X_train_df.shape)
print(X_test_df.shape)
print(y_train_df.shape)
print(y_test_df.shape)
```

```
                                                   Document
25873669  <data_structures.document.Document object at 0...
24269870  <data_structures.document.Document object at 0...
17655245  <data_structures.document.Document object at 0...
16663649  <data_structures.document.Document object at 0...
33907682  <data_structures.document.Document object at 0...
...                                                     ...
32079399  <data_structures.document.Document object at 0...
26976595  <data_structures.document.Document object at 0...
20138774  <data_structures.document.Document object at 0...
27936616  <data_structures.document.Document object at 0...
28009095  <data_structures.document.Document object at 0...

[606 rows x 1 columns]
                                                   Document
25873669  <data_structures.document.Document object at 0...
24269870  <data_structures.document.Document object at 0...
17655245  <data_structures.document.Document object at 0...
16663649  <data_structures.document.Document object at 0...
33907682  <data_structures.document.Document object at 0...
...                                                     ...
32079399  <data_structures.document.Document object at 0...
26976595  <data_structures.document.Document object at 0...
20138774  <data_structures.document.Document object at 0...
27936616  <data_structures.document.Document object at 0...
28009095  <data_structures.document.Document object at 0...

[606 rows x 1 columns]
25873669    0
24269870    0
17655245    1
16663649    1
33907682    0
           ..
32079399    1
26976595    0
20138774    1
27936616    1
28009095    1
Name: Label, Length: 606, dtype: int64
genetic diversity stilbene metabolism vitis sylvestris .
stilbenes , important secondary metabolites grapevine , represent central phytoalexins therefore constitute imp
ortant element basal immunity . study , potential genetic variation vitis vinifera ssp . sylvestris , ancestor
cultivated grapevine , sought respect output stilbenes potential use resistance breeding . considerable variati
on stilbene inducibility identified v. vinifera ssp . sylvestris . genotypic differences abundance profiles sti
lbenes induced response uv - c pulse shown . two clusters stilbene 'chemovars ' emerged : one cluster showed qu
ick strong accumulation stilbenes , almost exclusively form non - glycosylated resveratrol viniferin , second c
luster accumulated fewer stilbenes relatively high proportions piceatannol glycosylated piceid . 86 genotypes ,
time dependence stilbene pattern observed : piceid , resveratrol , piceatannol accumulated earlier , whereas vi
niferins found later . observed genotypic differences stilbene accumulation preceded differential accumulation
transcripts chalcone synthase ( chs ) stilbene - related genes : phenylalanine ammonium lyase ( pal ) , stilben
e synthase ( stsy ) , resveratrol synthase ( rs ) . screen population respect susceptibility downy mildew grape
vine ( plasmopara viticola ) revealed considerable variability . subpopulation genotypes high stilbene inducibi
lity significantly less susceptible compared low - stilbene genotypes , representative genotypes could shown in
ducibility stilbene synthase uv correlated inducibility pathogen .
(424, 1)
(182, 1)
(424,)
(182,)
```

In [29]:
```python
model_name = "HA_LSTM"
dl_config = DLConfig(model_name=model_name, seed_value=seed_value)
dl_config.stop_words = stops
dl_config.lower = True
dl_config.remove_punctuation = False
dl_config.split_by_hyphen = True
dl_config.lemmatization = False
dl_config.stems = False

#Parameters
dl_config.padding = 'post'            #'pre' -> default; 'post' -> alternative
dl_config.truncating = 'post'         #'pre' -> default; 'post' -> alternative      #####
dl_config.oov_token = 'OOV'

dl_config.epochs = 50
dl_config.batch_size = 32     # e aumentar o batch
dl_config.learning_rate = 0.001   #experimentar diminuir

dl_config.max_sent_len = 50      #sentences will have a maximum of "max_sent_len" words    #400/500
dl_config.max_nb_words = 100_000      #it will only be considered the top "max_nb_words" words in the dataset
dl_config.max_nb_sentences = 15    # set only for the hierarchical attention model!!!

dl_config.embeddings = 'biowordvec'

# igual ao modelo anterior
dl_config.embedding_path = './embeddings/biowordvec'
dl_config.embedding_dim = 200
dl_config.embedding_format = 'word2vec'

dl_config.keras_callbacks = True

 # compara as losses d otrain e validation
if dl_config.keras_callbacks:
    dl_config.patience = 5   #early-stopping patience
    checkpoint_path = str(dl_config.model_id_path) + '\checkpoint.hdf5'
    keras_callbacks = [
            EarlyStopping(monitor='val_loss', mode='min', verbose=1, patience=dl_config.patience),
           ModelCheckpoint(checkpoint_path, monitor='val_loss', mode='min', verbose=1, save_best_only=True)   #
    ]
else:
    keras_callbacks = None
```

In [ ]:

In [30]:
```python
dl_config.tokenizer = text.Tokenizer(num_words=dl_config.max_nb_words, oov_token=dl_config.oov_token)

dl_config.validation_percentage = 30  # talvez aumentar
X_train, y_train, x_val, y_val = DL_preprocessing(X_train_df, y_train_df,
                                                  dl_config, dataset='train',
                                                  validation_percentage = dl_config.validation_percentage,
                                                  seed_value=dl_config.seed_value)
X_train[0][0]
```

```
Found 7907 unique tokens.
Index of Unknown Words: 1
Training set with 297 samples
Validation set with 127 samples
```

Out[30]:
```
array([  22,    5,   20,  698, 1122, 3434,  178,   20,  698,  431,  497,
        333, 4734,  164,   86,    1,    0,    0,    0,    0,    0,    0,
          0,    0,    0,    0,    0,    0,    0,    0,    0,    0,    0,
          0,    0,    0,    0,    0,    0,    0,    0,    0,    0,    0,
          0,    0,    0,    0,    0,    0])
```

In [31]:
```python
dl_config.embedding_matrix = compute_embedding_matrix(dl_config, embeddings_format = dl_config.embedding_format
# quantas palavras encontrou no vocabulario do embedding

print(dl_config.embedding_matrix.shape)
print(dl_config.embedding_matrix)
```

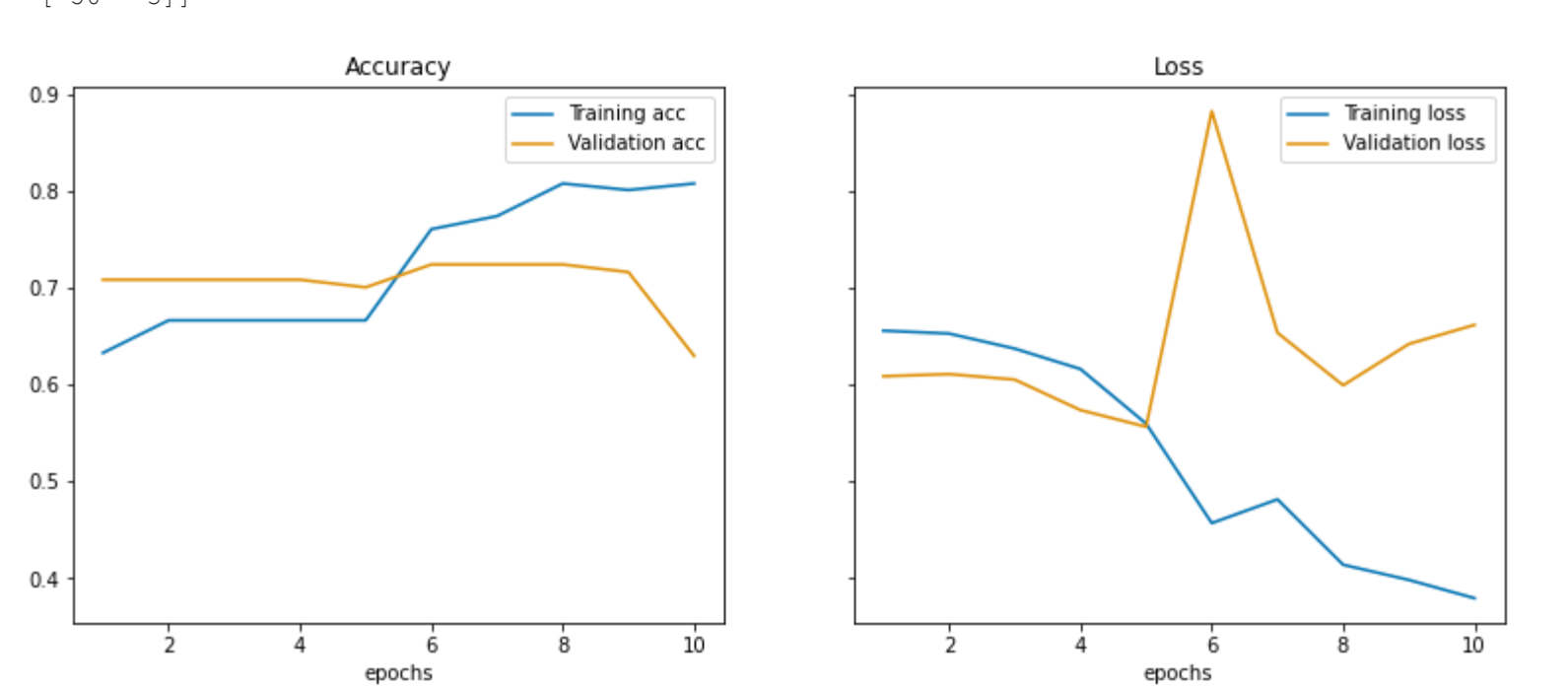
```
Creating Embedding Matrix...
Embedding Matrix Created
------------------------
number of null word embeddings: 350 in a total of 7907 words (4.43%)
words not found: 349
e.g. 10 words not found in the index :  ['hydroxyferuloyl' "'superior" "'jingzaojing" 'dk18' 'pholidotol'
 'dhs2001' '882000m' 'furtherstudy' 'metabolitism' 'piceides' '6β'
 'danfeng' 'myba2' 'sts36' 'spirotropis']
(7908, 200)
[[ 0.          0.          0.         ...  0.          0.
   0.        ]
 [ 0.          0.          0.         ...  0.          0.
   0.        ]
 [-0.0848686  -0.0417668  -0.3709338  ...  0.07843895  0.08479623
   0.54163074]
 ...
 [ 0.15182443 -0.13880545  0.16630234 ... -0.29269406 -0.08847184
  -0.05204243]
 [-0.1121414  -0.08872321 -0.33860227 ... -0.34313729  0.25840917
  -0.20746794]
 [ 0.26528463  0.19522475 -0.05939044 ...  0.18617576 -0.05572284
  -0.12646711]]
```

In [32]:
```python
if multiple_gpus:
    with strategy.scope():
        model = Hierarchical_Attention_LSTM(dl_config.embedding_matrix, dl_config, learning_rate=dl_config.lear
                                               seed_value=dl_config.seed_value)
else:
    model = Hierarchical_Attention_LSTM(dl_config.embedding_matrix, dl_config, learning_rate=dl_config.learning
                                               seed_value=dl_config.seed_value)


history = model.fit(X_train, y_train,
                    epochs=dl_config.epochs,
                    batch_size=dl_config.batch_size,
                    validation_data=(x_val,y_val),
                    callbacks=keras_callbacks)

if dl_config.keras_callbacks:
    model.load_weights(checkpoint_path)
```

```
Model: "model_4"
_________________________________________________________________
Layer (type)                 Output Shape              Param #
=================================================================
word_input (InputLayer)      [(None, 50)]              0
_________________________________________________________________
word_embedding (Embedding)   (None, 50, 200)           1581600
_________________________________________________________________
word_gru (Bidirectional)     (None, 50, 100)           100400
_________________________________________________________________
word_dense (Dense)           (None, 50, 100)           10100
_________________________________________________________________
word_attention (AttentionLay [(None, 100), (None, 50,  20400
=================================================================
Total params: 1,712,500
Trainable params: 130,900
Non-trainable params: 1,581,600
_________________________________________________________________
None
Model: "model_5"
_________________________________________________________________
Layer (type)                 Output Shape              Param #
=================================================================
sent_input (InputLayer)      [(None, 15, 50)]          0
_________________________________________________________________
sent_linking (TimeDistribute (None, 15, 100)           1712500
_________________________________________________________________
sent_gru (Bidirectional)     (None, 15, 100)           60400
_________________________________________________________________
dropout (Dropout)            (None, 15, 100)           0
_________________________________________________________________
sent_dense (Dense)           (None, 15, 100)           10100
_________________________________________________________________
sent_attention (AttentionLay [(None, 100), (None, 15,  20400
_________________________________________________________________
sent_dropout (Dropout)       (None, 100)               0
_________________________________________________________________
output (Dense)               (None, 1)                 101
=================================================================
Total params: 1,803,501
Trainable params: 221,901
Non-trainable params: 1,581,600
_________________________________________________________________
None
Epoch 1/50
10/10 [==============================] - 56s 1s/step - loss: 0.6769 - accuracy: 0.5541 - val_loss: 0.6089 - val
_accuracy: 0.708
7□□□□□□□□□□□□□□□□□□□□


Epoch 00001: val_loss improved from inf to 0.60891, saving model to models\HA_LSTM\HA_LSTM_0\checkpoint.hdf5
Epoch 2/50
10/10 [==============================] - 6s 608ms/step - loss: 0.6719 - accuracy: 0.6586 - val_loss: 0.6111 - v
al_accuracy: 0.708
7□□□□□□□□□□□□□□□□□□□□


Epoch 00002: val_loss did not improve from 0.60891
Epoch 3/50
10/10 [==============================] - 6s 614ms/step - loss: 0.6352 - accuracy: 0.6610 - val_loss: 0.6054 - v
al_accuracy: 0.708
7□□□□□□□□□□□□□□□□□□□□


Epoch 00003: val_loss improved from 0.60891 to 0.60544, saving model to models\HA_LSTM\HA_LSTM_0\checkpoint.hdf
5
Epoch 4/50
10/10 [==============================] - 7s 701ms/step - loss: 0.6026 - accuracy: 0.6876 - val_loss: 0.5739 - v
al_accuracy: 0.708
7□□□□□□□□□□□□□□□□□□□□


Epoch 00004: val_loss improved from 0.60544 to 0.57395, saving model to models\HA_LSTM\HA_LSTM_0\checkpoint.hdf
5
Epoch 5/50
10/10 [==============================] - 7s 673ms/step - loss: 0.5768 - accuracy: 0.6531 - val_loss: 0.5567 - v
al_accuracy: 0.700
8□□□□□□□□□□□□□□□□□□□□


Epoch 00005: val_loss improved from 0.57395 to 0.55669, saving model to models\HA_LSTM\HA_LSTM_0\checkpoint.hdf
5
Epoch 6/50
10/10 [==============================] - 7s 690ms/step - loss: 0.5097 - accuracy: 0.7159 - val_loss: 0.8829 - v
al_accuracy: 0.724
4□□□□□□□□□□□□□□□□□□□□


Epoch 00006: val_loss did not improve from 0.55669
Epoch 7/50
10/10 [==============================] - 6s 641ms/step - loss: 0.5229 - accuracy: 0.7595 - val_loss: 0.6538 - v
al_accuracy: 0.724
4□□□□□□□□□□□□□□□□□□□□


Epoch 00007: val_loss did not improve from 0.55669
Epoch 8/50
10/10 [==============================] - 7s 681ms/step - loss: 0.4294 - accuracy: 0.7785 - val_loss: 0.5995 - v
al_accuracy: 0.724
4□□□□□□□□□□□□□□□□□□□□


Epoch 00008: val_loss did not improve from 0.55669
Epoch 9/50
10/10 [==============================] - 6s 637ms/step - loss: 0.3833 - accuracy: 0.7948 - val_loss: 0.6423 - v
al_accuracy: 0.716
5□□□□□□□□□□□□□□□□□□□□


Epoch 00009: val_loss did not improve from 0.55669
Epoch 10/50
10/10 [==============================] - 7s 708ms/step - loss: 0.3938 - accuracy: 0.7814 - val_loss: 0.6620 - v
al_accuracy: 0.629
9□□□□□□□□□□□□□□□□□□□□


Epoch 00010: val_loss did not improve from 0.55669
Epoch 00010: early stopping
```

In [33]:
```python
train_loss, dl_config.train_acc = model.evaluate(X_train, y_train, verbose=0, batch_size = dl_config.batch_size
plot_training_history(history_dict=history, dl_config=dl_config)

x_test, y_test = DL_preprocessing(X_test_df, y_test_df, dl_config, dataset='test')

yhat_probs = model.predict(x_test, verbose=0)
yhat_probs = yhat_probs[:, 0]

yhat_classes = np.where(yhat_probs > 0.5, 1, yhat_probs)
yhat_classes = np.where(yhat_classes < 0.5, 0, yhat_classes).astype(np.int64)

dl_config.test_roc_auc, dl_config.test_pr_auc = plot_roc_n_pr_curves(y_test, yhat_probs, dl_config=dl_config)

# ROC AUC
print('ROC AUC: %f' % dl_config.test_roc_auc)

# avg precision
dl_config.test_avg_prec = average_precision(y_test_df, yhat_probs)
print('Average Precision: %f' % dl_config.test_avg_prec)

# accuracy
dl_config.test_acc = accuracy_score(y_test, yhat_classes)
print('Accuracy: %f' % dl_config.test_acc)

# precision tp / (tp + fp)
dl_config.test_prec = precision_score(y_test, yhat_classes)
print('Precision: %f' % dl_config.test_prec)

# recall: tp / (tp + fn)
dl_config.test_recall = recall_score(y_test, yhat_classes)
print('Recall: %f' % dl_config.test_recall)

# f1: 2 tp / (2 tp + fp + fn)
dl_config.test_f1_score = f1_score(y_test, yhat_classes)
print('F1 score: %f' % dl_config.test_f1_score)

# confusion matrix
matrix = confusion_matrix(y_test, yhat_classes)
matrix = np.flip(matrix)
print('Confusion Matrix:\n %s \n' % matrix)
```

```
Index of Unknown Words: 1
ROC AUC: 0.857792
Average Precision: 0.748001
Accuracy: 0.686813
Precision: 0.750000
Recall: 0.050847
F1 score: 0.095238
Confusion Matrix:
 [[122   1]
 [ 56   3]]
```



Burns_CNNBiLSTM

In [34]:
```python
"""Imports"""

from dl_models import Burns_CNNBiLSTM
```

In [35]:
```python
dl_config = DLConfig(remove_punctuation="HTML", stop_words=stops, lower=True, split_by_hyphen=False)

df = pandas.read_excel("C:/Users/Zé Freitas/Desktop/Mestrado/2°Semestre/Projeto/code/datasets/dataset_final.xls
                    index_col=0)
idsfinal = pandas_column_aslist(df, "Document")
docsfinal = pmids_to_docs(idsfinal, "pg42872@alunos.uminho.pt", dl_config)
docsfinal = docsfinal[0]
print(docsfinal[0].abstract_string)
dataset_docs = docs_to_pandasdocs(docsfinal)

x_total = dataset_docs
y_total = pandas.read_excel("C:/Users/Zé Freitas/Desktop/Mestrado/2°Semestre/Projeto/code/datasets/dataset_fina
                         index_col=1, usecols="B:F")  # os indices sao os IDs dos documentos

print(x_total)

relevance = pandas_column_aslist(y_total, "Relevance")
y_total = relevances_to_pandas(y_total, relevance)
print(y_total)

print(x_total["Document"][0].title_string)
print(x_total["Document"][0].abstract_string)

X_train_df, X_test_df, y_train_df, y_test_df = train_test_split(x_total, y_total, test_size=0.3, random_state=
                                                              stratify=y_total)
print(X_train_df.shape)
print(X_test_df.shape)
print(y_train_df.shape)
print(y_test_df.shape)
```

```
stilbenes , important secondary metabolites grapevine , represent central phytoalexins therefore constitute imp
ortant element basal immunity . study , potential genetic variation vitis vinifera ssp . sylvestris , ancestor
cultivated grapevine , sought respect output stilbenes potential use resistance breeding . considerable variati
on stilbene inducibility identified v. vinifera ssp . sylvestris . genotypic differences abundance profiles sti
lbenes induced response uv-c pulse shown . two clusters stilbene 'chemovars ' emerged : one cluster showed quic
k strong accumulation stilbenes , almost exclusively form non-glycosylated resveratrol viniferin , second clust
er accumulated fewer stilbenes relatively high proportions piceatannol glycosylated piceid . 86 genotypes , tim
e dependence stilbene pattern observed : piceid , resveratrol , piceatannol accumulated earlier , whereas vinif
erins found later . observed genotypic differences stilbene accumulation preceded differential accumulation tra
nscripts chalcone synthase ( chs ) stilbene-related genes : phenylalanine ammonium lyase ( pal ) , stilbene syn
thase ( stsy ) , resveratrol synthase ( rs ) . screen population respect susceptibility downy mildew grapevine
( plasmopara viticola ) revealed considerable variability . subpopulation genotypes high stilbene inducibility
significantly less susceptible compared low-stilbene genotypes , representative genotypes could shown inducibil
ity stilbene synthase uv correlated inducibility pathogen .
                                                   Document
25873669  <data_structures.document.Document object at 0...
24269870  <data_structures.document.Document object at 0...
17655245  <data_structures.document.Document object at 0...
16663649  <data_structures.document.Document object at 0...
33907682  <data_structures.document.Document object at 0...
...                                                     ...
32079399  <data_structures.document.Document object at 0...
26976595  <data_structures.document.Document object at 0...
20138774  <data_structures.document.Document object at 0...
27936616  <data_structures.document.Document object at 0...
28009095  <data_structures.document.Document object at 0...

[606 rows x 1 columns]
25873669    0
24269870    0
17655245    1
16663649    1
33907682    0
           ..
32079399    1
26976595    0
20138774    1
27936616    1
28009095    1
Name: Label, Length: 606, dtype: int64
genetic diversity stilbene metabolism vitis sylvestris .
stilbenes , important secondary metabolites grapevine , represent central phytoalexins therefore constitute imp
ortant element basal immunity . study , potential genetic variation vitis vinifera ssp . sylvestris , ancestor
cultivated grapevine , sought respect output stilbenes potential use resistance breeding . considerable variati
on stilbene inducibility identified v. vinifera ssp . sylvestris . genotypic differences abundance profiles sti
lbenes induced response uv-c pulse shown . two clusters stilbene 'chemovars ' emerged : one cluster showed quic
k strong accumulation stilbenes , almost exclusively form non-glycosylated resveratrol viniferin , second clust
er accumulated fewer stilbenes relatively high proportions piceatannol glycosylated piceid . 86 genotypes , tim
e dependence stilbene pattern observed : piceid , resveratrol , piceatannol accumulated earlier , whereas vinif
erins found later . observed genotypic differences stilbene accumulation preceded differential accumulation tra
nscripts chalcone synthase ( chs ) stilbene-related genes : phenylalanine ammonium lyase ( pal ) , stilbene syn
thase ( stsy ) , resveratrol synthase ( rs ) . screen population respect susceptibility downy mildew grapevine
( plasmopara viticola ) revealed considerable variability . subpopulation genotypes high stilbene inducibility
significantly less susceptible compared low-stilbene genotypes , representative genotypes could shown inducibil
ity stilbene synthase uv correlated inducibility pathogen .
(424, 1)
(182, 1)
(424,)
(182,)
```

In [37]:
```python
model_name = "BURNS"
dl_config = DLConfig(model_name=model_name, seed_value=seed_value)
dl_config.stop_words = stops
dl_config.lower = True
dl_config.remove_punctuation = False
dl_config.split_by_hyphen = False
dl_config.lemmatization = False
dl_config.stems = True

dl_config.padding = 'post'            #'pre' -> default; 'post' -> alternative
dl_config.truncating = 'post'         #'pre' -> default; 'post' -> alternative      #####
dl_config.oov_token = 'OOV'

dl_config.epochs = 50
dl_config.batch_size = 32     # e aumentar o batch
dl_config.learning_rate = 0.001   #experimentar diminuir

dl_config.max_sent_len = 300      #sentences will have a maximum of "max_sent_len" words    #400/500
dl_config.max_nb_words = 100_000      #it will only be considered the top "max_nb_words" words in the dataset

dl_config.embeddings = 'biowordvec'

dl_config.embedding_path = './embeddings/biowordvec'
dl_config.embedding_dim = 200
dl_config.embedding_format = 'word2vec'

dl_config.keras_callbacks = True

 # compara as losses d otrain e validation
if dl_config.keras_callbacks:
    dl_config.patience = 5   #early-stopping patience
    checkpoint_path = str(dl_config.model_id_path) + '\checkpoint.hdf5'
    keras_callbacks = [
            EarlyStopping(monitor='val_loss', mode='min', verbose=1, patience=dl_config.patience),
           ModelCheckpoint(checkpoint_path, monitor='val_loss', mode='min', verbose=1, save_best_only=True)   #
    ]
else:
    keras_callbacks = None
```

In [38]:
```python
dl_config.tokenizer = text.Tokenizer(num_words=dl_config.max_nb_words, oov_token=dl_config.oov_token)

dl_config.validation_percentage = 30  # talvez aumentar
X_train, y_train, x_val, y_val = DL_preprocessing(X_train_df, y_train_df,
                                                  dl_config, dataset='train',
                                                  validation_percentage=dl_config.validation_percentage,
                                                  seed_value=dl_config.seed_value)
X_train[0][0]
```

```
Found 7907 unique tokens.
Training set with 297 samples
Validation set with 127 samples
```

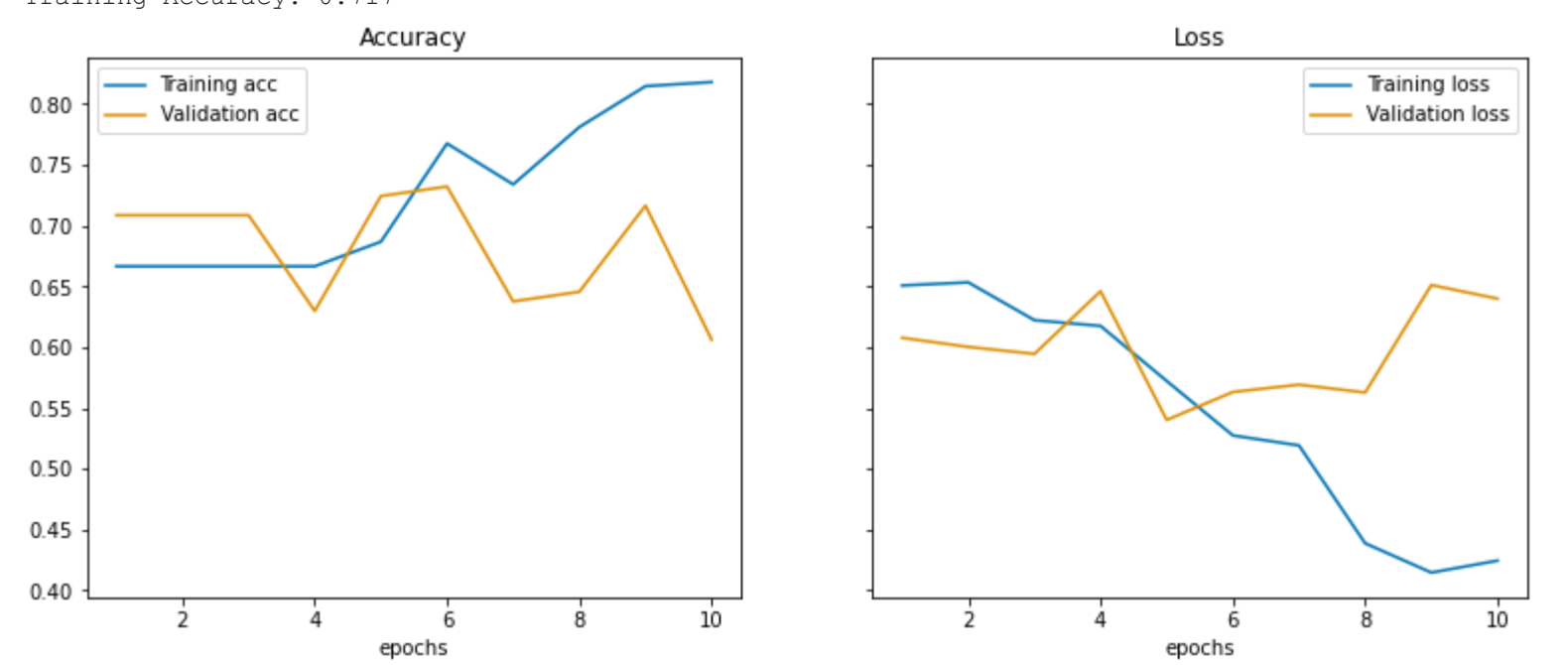
Out[38]: 22

In [39]:
```python
dl_config.embedding_matrix = compute_embedding_matrix(dl_config, embeddings_format = dl_config.embedding_format

print(dl_config.embedding_matrix.shape)
print(dl_config.embedding_matrix)
```

```
Creating Embedding Matrix...
Embedding Matrix Created
------------------------
number of null word embeddings: 350 in a total of 7907 words (4.43%)
words not found: 349
e.g. 10 words not found in the index :  ['hydroxyferuloyl' "'superior" "'jingzaojing" 'dk18' 'pholidotol'
 'dhs2001' '882000m' 'furtherstudy' 'metabolitism' 'piceides' '6β'
 'danfeng' 'myba2' 'sts36' 'spirotropis']
(7908, 200)
[[ 0.          0.          0.         ...  0.          0.
   0.        ]
 [ 0.          0.          0.         ...  0.          0.
   0.        ]
 [-0.0848686  -0.0417668  -0.3709338  ...  0.07843895  0.08479623
   0.54163074]
 ...
 [ 0.15182443 -0.13880545  0.16630234 ... -0.29269406 -0.08847184
  -0.05204243]
 [-0.1121414  -0.08872321 -0.33860227 ... -0.34313729  0.25840917
  -0.20746794]
 [ 0.26528463  0.19522475 -0.05939044 ...  0.18617576 -0.05572284
  -0.12646711]]
```

In [40]:
```python
if multiple_gpus:
    with strategy.scope():
        model = Burns_CNNBiLSTM(dl_config.embedding_matrix, dl_config, learning_rate=dl_config.learning_rate,
                                               seed_value=dl_config.seed_value)
else:
    model = Burns_CNNBiLSTM(dl_config.embedding_matrix, dl_config, learning_rate=dl_config.learning_rate,
                                               seed_value=dl_config.seed_value)


history = model.fit(X_train, y_train,
                    epochs=dl_config.epochs,
                    batch_size=dl_config.batch_size,
                    validation_data=(x_val,y_val),
                    callbacks=keras_callbacks)

if dl_config.keras_callbacks:
    model.load_weights(checkpoint_path)

train_loss, dl_config.train_acc = model.evaluate(X_train, y_train, verbose=0)

print('Training Loss: %.3f' % (train_loss))
print('Training Accuracy: %.3f' % (dl_config.train_acc))

plot_training_history(history_dict = history, dl_config=dl_config)

x_test, y_test = DL_preprocessing(X_test_df, y_test_df, dl_config, dataset='test')

yhat_probs = model.predict(x_test, verbose=0)
yhat_probs = yhat_probs[:, 0]

yhat_classes = np.where(yhat_probs > 0.5, 1, yhat_probs)
yhat_classes = np.where(yhat_classes < 0.5, 0, yhat_classes).astype(np.int64)
```

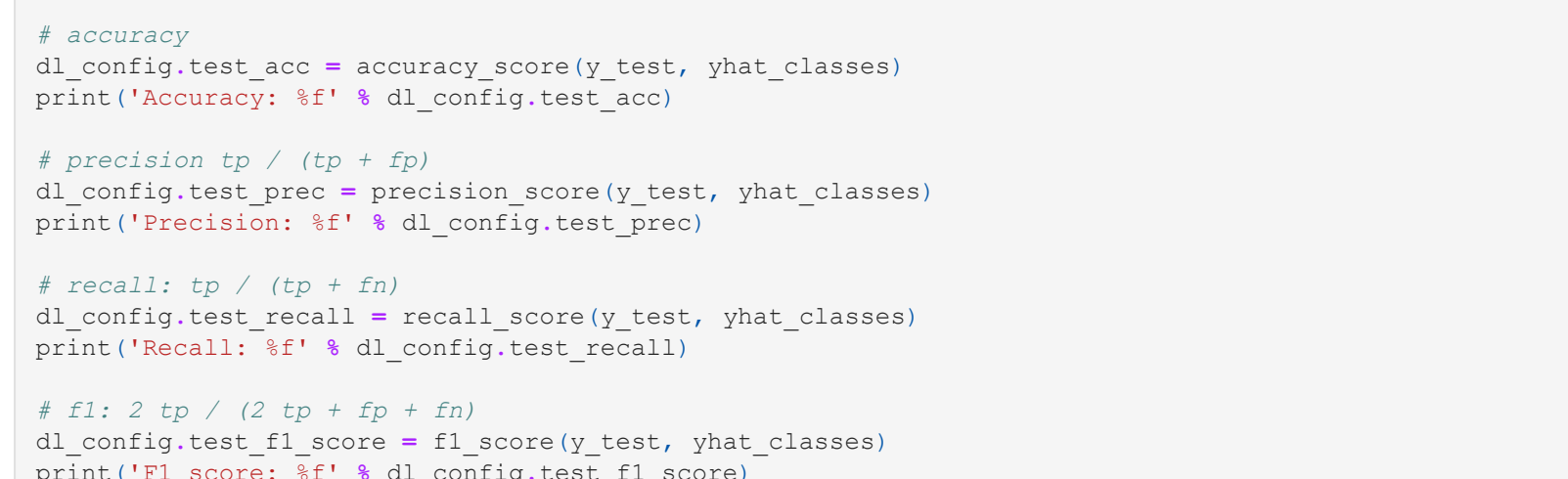
```
Model: "model_6"
__________________________________________________________________________________________________
Layer (type)                    Output Shape         Param #     Connected to
==================================================================================================
input_5 (InputLayer)            [(None, 300)]        0
__________________________________________________________________________________________________
embedding (Embedding)           (None, 300, 200)     1581600     input_5[0][0]
__________________________________________________________________________________________________
dropout_1 (Dropout)             (None, 300, 200)     0           embedding[0][0]
__________________________________________________________________________________________________
conv1d (Conv1D)                 (None, 300, 16)      16016       dropout_1[0][0]
__________________________________________________________________________________________________
max_pooling1d (MaxPooling1D)    (None, 75, 16)       0           conv1d[0][0]
__________________________________________________________________________________________________
conv1d_1 (Conv1D)               (None, 75, 16)       1296        max_pooling1d[0][0]
__________________________________________________________________________________________________
dropout_2 (Dropout)             (None, 75, 16)       0           conv1d_1[0][0]
__________________________________________________________________________________________________
bidirectional_4 (Bidirectional) (None, 128)          41472       dropout_2[0][0]
__________________________________________________________________________________________________
dropout_3 (Dropout)             (None, 128)          0           bidirectional_4[0][0]
__________________________________________________________________________________________________
dense_2 (Dense)                 (None, 1)            129         dropout_3[0][0]
__________________________________________________________________________________________________
flatten (Flatten)               (None, 1)            0           dense_2[0][0]
__________________________________________________________________________________________________
activation (Activation)         (None, 1)            0           flatten[0][0]
__________________________________________________________________________________________________
repeat_vector (RepeatVector)    (None, 128, 1)       0           activation[0][0]
__________________________________________________________________________________________________
permute (Permute)               (None, 1, 128)       0           repeat_vector[0][0]
__________________________________________________________________________________________________
multiply (Multiply)             (None, 1, 128)       0           dropout_3[0][0]
                                                                 permute[0][0]
__________________________________________________________________________________________________
lambda (Lambda)                 (None, 128)          0           multiply[0][0]
__________________________________________________________________________________________________
dense_3 (Dense)                 (None, 1)            129         lambda[0][0]
==================================================================================================
Total params: 1,640,642
Trainable params: 59,042
Non-trainable params: 1,581,600
__________________________________________________________________________________________________
Epoch 1/50
10/10 [==============================] - 68s 5s/step - loss: 0.6748 - accuracy: 0.6406 - val_loss: 0.6078 - val
_accuracy: 0.708
7□□□□□□□□□□□□□□□□□□□□□□□□□□□□□□□□□□□□□□□□□□□□□□□□□□□□□□□□□□□□□□□□□□□□□□□


Epoch 00001: val_loss improved from inf to 0.60784, saving model to models\BURNS\BURNS_0\checkpoint.hdf5
Epoch 2/50
10/10 [==============================] - 2s 199ms/step - loss: 0.6622 - accuracy: 0.6586 - val_loss: 0.6003 - v
al_accuracy: 0.708
7□□□□□□□□□□□□□□□□□□□□□□□□□□□□□□□□□□□□□□□□□□□□□□□□□□□□□□□□□□□□□□□□□□□□□□□


Epoch 00002: val_loss improved from 0.60784 to 0.60034, saving model to models\BURNS\BURNS_0\checkpoint.hdf5
Epoch 3/50
10/10 [==============================] - 2s 196ms/step - loss: 0.6312 - accuracy: 0.6610 - val_loss: 0.5946 - v
al_accuracy: 0.708
7□□□□□□□□□□□□□□□□□□□□□□□□□□□□□□□□□□□□□□□□□□□□□□□□□□□□□□□□□□□□□□□□□□□□□□□


Epoch 00003: val_loss improved from 0.60034 to 0.59462, saving model to models\BURNS\BURNS_0\checkpoint.hdf5
Epoch 4/50
10/10 [==============================] - 2s 196ms/step - loss: 0.5870 - accuracy: 0.6909 - val_loss: 0.6461 - v
al_accuracy: 0.629
9□□□□□□□□□□□□□□□□□□□□□□□□□□□□□□□□□□□□□□□□□□□□□□□□□□□□□□□□□□□□□□□□□□□□□□□


Epoch 00004: val_loss did not improve from 0.59462
Epoch 5/50
10/10 [==============================] - 2s 194ms/step - loss: 0.5876 - accuracy: 0.6710 - val_loss: 0.5403 - v
al_accuracy: 0.724
4□□□□□□□□□□□□□□□□□□□□□□□□□□□□□□□□□□□□□□□□□□□□□□□□□□□□□□□□□□□□□□□□□□□□□□□


Epoch 00005: val_loss improved from 0.59462 to 0.54026, saving model to models\BURNS\BURNS_0\checkpoint.hdf5
Epoch 6/50
10/10 [==============================] - 2s 191ms/step - loss: 0.5566 - accuracy: 0.7338 - val_loss: 0.5633 - v
al_accuracy: 0.732
3□□□□□□□□□□□□□□□□□□□□□□□□□□□□□□□□□□□□□□□□□□□□□□□□□□□□□□□□□□□□□□□□□□□□□□□


Epoch 00006: val_loss did not improve from 0.54026
Epoch 7/50
10/10 [==============================] - 2s 194ms/step - loss: 0.5304 - accuracy: 0.7111 - val_loss: 0.5693 - v
al_accuracy: 0.637
8□□□□□□□□□□□□□□□□□□□□□□□□□□□□□□□□□□□□□□□□□□□□□□□□□□□□□□□□□□□□□□□□□□□□□□□


Epoch 00007: val_loss did not improve from 0.54026
Epoch 8/50
10/10 [==============================] - 2s 196ms/step - loss: 0.4354 - accuracy: 0.7922 - val_loss: 0.5628 - v
al_accuracy: 0.645
7□□□□□□□□□□□□□□□□□□□□□□□□□□□□□□□□□□□□□□□□□□□□□□□□□□□□□□□□□□□□□□□□□□□□□□□


Epoch 00008: val_loss did not improve from 0.54026
Epoch 9/50
10/10 [==============================] - 2s 198ms/step - loss: 0.4167 - accuracy: 0.8163 - val_loss: 0.6512 - v
al_accuracy: 0.716
5□□□□□□□□□□□□□□□□□□□□□□□□□□□□□□□□□□□□□□□□□□□□□□□□□□□□□□□□□□□□□□□□□□□□□□□


Epoch 00009: val_loss did not improve from 0.54026
Epoch 10/50
10/10 [==============================] - 2s 195ms/step - loss: 0.4372 - accuracy: 0.8173 - val_loss: 0.6400 - v
al_accuracy: 0.606
3□□□□□□□□□□□□□□□□□□□□□□□□□□□□□□□□□□□□□□□□□□□□□□□□□□□□□□□□□□□□□□□□□□□□□□□


Epoch 00010: val_loss did not improve from 0.54026
Epoch 00010: early stopping
Training Loss: 0.511
Training Accuracy: 0.717
```



In [41]:

```python
dl_config.test_roc_auc, dl_config.test_pr_auc = plot_roc_n_pr_curves(y_test, yhat_probs, dl_config=dl_config)

# ROC AUC
print('ROC AUC: %f' % dl_config.test_roc_auc)

# avg precision
dl_config.test_avg_prec = average_precision(y_test_df, yhat_probs)
print('Average Precision: %f' % dl_config.test_avg_prec)

# accuracy
dl_config.test_acc = accuracy_score(y_test, yhat_classes)
print('Accuracy: %f' % dl_config.test_acc)

# precision tp / (tp + fp)
dl_config.test_prec = precision_score(y_test, yhat_classes)
print('Precision: %f' % dl_config.test_prec)

# recall: tp / (tp + fn)
dl_config.test_recall = recall_score(y_test, yhat_classes)
print('Recall: %f' % dl_config.test_recall)

# f1: 2 tp / (2 tp + fp + fn)
dl_config.test_f1_score = f1_score(y_test, yhat_classes)
print('F1 score: %f' % dl_config.test_f1_score)

# confusion matrix
matrix = confusion_matrix(y_test, yhat_classes)
matrix = np.flip(matrix)
print('Confusion Matrix:\n %s \n' % matrix)
```

```
ROC AUC: 0.853521
Average Precision: 0.696908
Accuracy: 0.725275
Precision: 0.736842
Recall: 0.237288
F1 score: 0.358974
Confusion Matrix:
 [[118   5]
 [ 45  14]]
```



BioBert_LSTM

In [13]:

```python
seed_value = 42
model_name = "Bert_LSTM"
np.random.seed(seed_value)
random.seed(seed_value)
tensorflow.random.set_seed(seed_value)
print(sys.executable)

import torch
if seed_value:
    torch.manual_seed(seed_value)
    torch.cuda.manual_seed_all(seed_value)
    torch.backends.cudnn.deterministic = True
    torch.backends.cudnn.benchmark = False
os.environ['PYTHONHASHSEED'] = str(seed_value)
os.environ['TF_DETERMINISTIC_OPS'] = '1'

if multiple_gpus:
    devices = []
    for gpu in multiple_gpus:
        devices.append('/gpu:' + str(gpu))
    strategy = tensorflow.distribute.MirroredStrategy(devices=devices)
    os.environ["CUDA_VISIBLE_DEVICES"] = ''

else:
    # Get the GPU device name.
    device_name = tensorflow.test.gpu_device_name()
    # The device name should look like the following:
    if device_name == '/device:GPU:0':
        print('Using GPU: {}'.format(device_name))
    else:
        raise SystemError('GPU device not found')

    os.environ["CUDA_VISIBLE_DEVICES"] = device_name
    os.environ["CUDA_DEVICE_ORDER"] = "PCI_BUS_ID"
```

```
C:\ProgramData\Anaconda3\python.exe
WARNING:tensorflow:Some requested devices in `tf.distribute.Strategy` are not visible to TensorFlow: /job:local
host/replica:0/task:0/device:GPU:0
INFO:tensorflow:Using MirroredStrategy with devices ('/job:localhost/replica:0/task:0/device:GPU:0',)
```

In [14]:

```python
"""Imports"""

from dl import Bert_preprocessing
from dl_models import Bert_LSTM
from transformers import BertTokenizer

os.environ["WANDB_API_KEY"] = "0" ## to silence warning
!transformers-cli convert --model_type bert \
  --tf_checkpoint "C:/Users/Zé Freitas/Desktop/Mestrado/2°Semestre/Projeto/code/biobert_v1.1_pubmed/model.ckpt-
  --config "C:/Users/Zé Freitas/Desktop/Mestrado/2°Semestre/Projeto/code/biobert_v1.1_pubmed/bert_config.json"
  --pytorch_dump_output "C:/Users/Zé Freitas/Desktop/Mestrado/2°Semestre/Projeto/code/biobert_v1.1_pubmed/pyto
```

```
Building PyTorch model from configuration: BertConfig {
  "attention_probs_dropout_prob": 0.1,
  "gradient_checkpointing": false,
  "hidden_act": "gelu",
  "hidden_dropout_prob": 0.1,
  "hidden_size": 768,
  "initializer_range": 0.02,
  "intermediate_size": 3072,
  "layer_norm_eps": 1e-12,
  "max_position_embeddings": 512,
  "model_type": "bert",
  "num_attention_heads": 12,
  "num_hidden_layers": 12,
  "pad_token_id": 0,
  "position_embedding_type": "absolute",
  "transformers_version": "4.6.1",
  "type_vocab_size": 2,
  "use_cache": true,
  "vocab_size": 28996
}

Save PyTorch model to C:/Users/Zé Freitas/Desktop/Mestrado/2°Semestre/Projeto/code/biobert_v1.1_pubmed/pytorch_
model.bin
```

```
2021-06-22 19:00:48.749461: W tensorflow/stream_executor/platform/default/dso_loader.cc:60] Could not load dyna
mic library 'cudart64_110.dll'; dlerror: cudart64_110.dll not found
2021-06-22 19:00:48.749592: I tensorflow/stream_executor/cuda/cudart_stub.cc:29] Ignore above cudart dlerror if
you do not have a GPU set up on your machine.
Converting TensorFlow checkpoint from C:\Users\Zé Freitas\Desktop\Mestrado\2°Semestre\Projeto\code\biobert_v1.1
_pubmed\model.ckpt-1000000
Loading TF weight bert/embeddings/LayerNorm/beta with shape [768]
Loading TF weight bert/embeddings/LayerNorm/gamma with shape [768]
Loading TF weight bert/embeddings/position_embeddings with shape [512, 768]
Loading TF weight bert/embeddings/token_type_embeddings with shape [2, 768]
Loading TF weight bert/embeddings/word_embeddings with shape [28996, 768]
Loading TF weight bert/encoder/layer_0/attention/output/LayerNorm/beta with shape [768]
Loading TF weight bert/encoder/layer_0/attention/output/LayerNorm/gamma with shape [768]
Loading TF weight bert/encoder/layer_0/attention/output/dense/bias with shape [768]
Loading TF weight bert/encoder/layer_0/attention/output/dense/kernel with shape [768, 768]
Loading TF weight bert/encoder/layer_0/attention/self/key/bias with shape [768]
Loading TF weight bert/encoder/layer_0/attention/self/key/kernel with shape [768, 768]
Loading TF weight bert/encoder/layer_0/attention/self/query/bias with shape [768]
Loading TF weight bert/encoder/layer_0/attention/self/query/kernel with shape [768, 768]
Loading TF weight bert/encoder/layer_0/attention/self/value/bias with shape [768]
Loading TF weight bert/encoder/layer_0/attention/self/value/kernel with shape [768, 768]
Loading TF weight bert/encoder/layer_0/intermediate/dense/bias with shape [3072]
Loading TF weight bert/encoder/layer_0/intermediate/dense/kernel with shape [768, 3072]
Loading TF weight bert/encoder/layer_0/output/LayerNorm/beta with shape [768]
Loading TF weight bert/encoder/layer_0/output/LayerNorm/gamma with shape [768]
Loading TF weight bert/encoder/layer_0/output/dense/bias with shape [768]
Loading TF weight bert/encoder/layer_0/output/dense/kernel with shape [3072, 768]
Loading TF weight bert/encoder/layer_1/attention/output/LayerNorm/beta with shape [768]
Loading TF weight bert/encoder/layer_1/attention/output/LayerNorm/gamma with shape [768]
Loading TF weight bert/encoder/layer_1/attention/output/dense/bias with shape [768]
Loading TF weight bert/encoder/layer_1/attention/output/dense/kernel with shape [768, 768]
Loading TF weight bert/encoder/layer_1/attention/self/key/bias with shape [768]
Loading TF weight bert/encoder/layer_1/attention/self/key/kernel with shape [768, 768]
Loading TF weight bert/encoder/layer_1/attention/self/query/bias with shape [768]
Loading TF weight bert/encoder/layer_1/attention/self/query/kernel with shape [768, 768]
Loading TF weight bert/encoder/layer_1/attention/self/value/bias with shape [768]
Loading TF weight bert/encoder/layer_1/attention/self/value/kernel with shape [768, 768]
Loading TF weight bert/encoder/layer_1/intermediate/dense/bias with shape [3072]
Loading TF weight bert/encoder/layer_1/intermediate/dense/kernel with shape [768, 3072]
Loading TF weight bert/encoder/layer_1/output/LayerNorm/beta with shape [768]
Loading TF weight bert/encoder/layer_1/output/LayerNorm/gamma with shape [768]
Loading TF weight bert/encoder/layer_1/output/dense/bias with shape [768]
Loading TF weight bert/encoder/layer_1/output/dense/kernel with shape [3072, 768]
Loading TF weight bert/encoder/layer_10/attention/output/LayerNorm/beta with shape [768]
Loading TF weight bert/encoder/layer_10/attention/output/LayerNorm/gamma with shape [768]
Loading TF weight bert/encoder/layer_10/attention/output/dense/bias with shape [768]
Loading TF weight bert/encoder/layer_10/attention/output/dense/kernel with shape [768, 768]
Loading TF weight bert/encoder/layer_10/attention/self/key/bias with shape [768]
Loading TF weight bert/encoder/layer_10/attention/self/key/kernel with shape [768, 768]
Loading TF weight bert/encoder/layer_10/attention/self/query/bias with shape [768]
Loading TF weight bert/encoder/layer_10/attention/self/query/kernel with shape [768, 768]
Loading TF weight bert/encoder/layer_10/attention/self/value/bias with shape [768]
Loading TF weight bert/encoder/layer_10/attention/self/value/kernel with shape [768, 768]
Loading TF weight bert/encoder/layer_10/intermediate/dense/bias with shape [3072]
Loading TF weight bert/encoder/layer_10/intermediate/dense/kernel with shape [768, 3072]
Loading TF weight bert/encoder/layer_10/output/LayerNorm/beta with shape [768]
Loading TF weight bert/encoder/layer_10/output/LayerNorm/gamma with shape [768]
Loading TF weight bert/encoder/layer_10/output/dense/bias with shape [768]
Loading TF weight bert/encoder/layer_10/output/dense/kernel with shape [3072, 768]
Loading TF weight bert/encoder/layer_11/attention/output/LayerNorm/beta with shape [768]
Loading TF weight bert/encoder/layer_11/attention/output/LayerNorm/gamma with shape [768]
Loading TF weight bert/encoder/layer_11/attention/output/dense/bias with shape [768]
Loading TF weight bert/encoder/layer_11/attention/output/dense/kernel with shape [768, 768]
Loading TF weight bert/encoder/layer_11/attention/self/key/bias with shape [768]
Loading TF weight bert/encoder/layer_11/attention/self/key/kernel with shape [768, 768]
Loading TF weight bert/encoder/layer_11/attention/self/query/bias with shape [768]
Loading TF weight bert/encoder/layer_11/attention/self/query/kernel with shape [768, 768]
Loading TF weight bert/encoder/layer_11/attention/self/value/bias with shape [768]
Loading TF weight bert/encoder/layer_11/attention/self/value/kernel with shape [768, 768]
Loading TF weight bert/encoder/layer_11/intermediate/dense/bias with shape [3072]
Loading TF weight bert/encoder/layer_11/intermediate/dense/kernel with shape [768, 3072]
Loading TF weight bert/encoder/layer_11/output/LayerNorm/beta with shape [768]
Loading TF weight bert/encoder/layer_11/output/LayerNorm/gamma with shape [768]
Loading TF weight bert/encoder/layer_11/output/dense/bias with shape [768]
Loading TF weight bert/encoder/layer_11/output/dense/kernel with shape [3072, 768]
Loading TF weight bert/encoder/layer_2/attention/output/LayerNorm/beta with shape [768]
Loading TF weight bert/encoder/layer_2/attention/output/LayerNorm/gamma with shape [768]
Loading TF weight bert/encoder/layer_2/attention/output/dense/bias with shape [768]
Loading TF weight bert/encoder/layer_2/attention/output/dense/kernel with shape [768, 768]
Loading TF weight bert/encoder/layer_2/attention/self/key/bias with shape [768]
Loading TF weight bert/encoder/layer_2/attention/self/key/kernel with shape [768, 768]
Loading TF weight bert/encoder/layer_2/attention/self/query/bias with shape [768]
Loading TF weight bert/encoder/layer_2/attention/self/query/kernel with shape [768, 768]
Loading TF weight bert/encoder/layer_2/attention/self/value/bias with shape [768]
Loading TF weight bert/encoder/layer_2/attention/self/value/kernel with shape [768, 768]
Loading TF weight bert/encoder/layer_2/intermediate/dense/bias with shape [3072]
Loading TF weight bert/encoder/layer_2/intermediate/dense/kernel with shape [768, 3072]
Loading TF weight bert/encoder/layer_2/output/LayerNorm/beta with shape [768]
Loading TF weight bert/encoder/layer_2/output/LayerNorm/gamma with shape [768]
Loading TF weight bert/encoder/layer_2/output/dense/bias with shape [768]
Loading TF weight bert/encoder/layer_2/output/dense/kernel with shape [3072, 768]
Loading TF weight bert/encoder/layer_3/attention/output/LayerNorm/beta with shape [768]
Loading TF weight bert/encoder/layer_3/attention/output/LayerNorm/gamma with shape [768]
Loading TF weight bert/encoder/layer_3/attention/output/dense/bias with shape [768]
Loading TF weight bert/encoder/layer_3/attention/output/dense/kernel with shape [768, 768]
Loading TF weight bert/encoder/layer_3/attention/self/key/bias with shape [768]
Loading TF weight bert/encoder/layer_3/attention/self/key/kernel with shape [768, 768]
Loading TF weight bert/encoder/layer_3/attention/self/query/bias with shape [768]
Loading TF weight bert/encoder/layer_3/attention/self/query/kernel with shape [768, 768]
Loading TF weight bert/encoder/layer_3/attention/self/value/bias with shape [768]
Loading TF weight bert/encoder/layer_3/attention/self/value/kernel with shape [768, 768]
Loading TF weight bert/encoder/layer_3/intermediate/dense/bias with shape [3072]
Loading TF weight bert/encoder/layer_3/intermediate/dense/kernel with shape [768, 3072]
Loading TF weight bert/encoder/layer_3/output/LayerNorm/beta with shape [768]
Loading TF weight bert/encoder/layer_3/output/LayerNorm/gamma with shape [768]
Loading TF weight bert/encoder/layer_3/output/dense/bias with shape [768]
Loading TF weight bert/encoder/layer_3/output/dense/kernel with shape [3072, 768]
Loading TF weight bert/encoder/layer_4/attention/output/LayerNorm/beta with shape [768]
Loading TF weight bert/encoder/layer_4/attention/output/LayerNorm/gamma with shape [768]
Loading TF weight bert/encoder/layer_4/attention/output/dense/bias with shape [768]
Loading TF weight bert/encoder/layer_4/attention/output/dense/kernel with shape [768, 768]
Loading TF weight bert/encoder/layer_4/attention/self/key/bias with shape [768]
Loading TF weight bert/encoder/layer_4/attention/self/key/kernel with shape [768, 768]
Loading TF weight bert/encoder/layer_4/attention/self/query/bias with shape [768]
Loading TF weight bert/encoder/layer_4/attention/self/query/kernel with shape [768, 768]
Loading TF weight bert/encoder/layer_4/attention/self/value/bias with shape [768]
Loading TF weight bert/encoder/layer_4/attention/self/value/kernel with shape [768, 768]
Loading TF weight bert/encoder/layer_4/intermediate/dense/bias with shape [3072]
Loading TF weight bert/encoder/layer_4/intermediate/dense/kernel with shape [768, 3072]
Loading TF weight bert/encoder/layer_4/output/LayerNorm/beta with shape [768]
Loading TF weight bert/encoder/layer_4/output/LayerNorm/gamma with shape [768]
Loading TF weight bert/encoder/layer_4/output/dense/bias with shape [768]
Loading TF weight bert/encoder/layer_4/output/dense/kernel with shape [3072, 768]
Loading TF weight bert/encoder/layer_5/attention/output/LayerNorm/beta with shape [768]
Loading TF weight bert/encoder/layer_5/attention/output/LayerNorm/gamma with shape [768]
Loading TF weight bert/encoder/layer_5/attention/output/dense/bias with shape [768]
Loading TF weight bert/encoder/layer_5/attention/output/dense/kernel with shape [768, 768]
Loading TF weight bert/encoder/layer_5/attention/self/key/bias with shape [768]
Loading TF weight bert/encoder/layer_5/attention/self/key/kernel with shape [768, 768]
Loading TF weight bert/encoder/layer_5/attention/self/query/bias with shape [768]
Loading TF weight bert/encoder/layer_5/attention/self/query/kernel with shape [768, 768]
Loading TF weight bert/encoder/layer_5/attention/self/value/bias with shape [768]
Loading TF weight bert/encoder/layer_5/attention/self/value/kernel with shape [768, 768]
Loading TF weight bert/encoder/layer_5/intermediate/dense/bias with shape [3072]
Loading TF weight bert/encoder/layer_5/intermediate/dense/kernel with shape [768, 3072]
Loading TF weight bert/encoder/layer_5/output/LayerNorm/beta with shape [768]
Loading TF weight bert/encoder/layer_5/output/LayerNorm/gamma with shape [768]
Loading TF weight bert/encoder/layer_5/output/dense/bias with shape [768]
Loading TF weight bert/encoder/layer_5/output/dense/kernel with shape [3072, 768]
Loading TF weight bert/encoder/layer_6/attention/output/LayerNorm/beta with shape [768]
Loading TF weight bert/encoder/layer_6/attention/output/LayerNorm/gamma with shape [768]
Loading TF weight bert/encoder/layer_6/attention/output/dense/bias with shape [768]
Loading TF weight bert/encoder/layer_6/attention/output/dense/kernel with shape [768, 768]
Loading TF weight bert/encoder/layer_6/attention/self/key/bias with shape [768]
Loading TF weight bert/encoder/layer_6/attention/self/key/kernel with shape [768, 768]
Loading TF weight bert/encoder/layer_6/attention/self/query/bias with shape [768]
Loading TF weight bert/encoder/layer_6/attention/self/query/kernel with shape [768, 768]
Loading TF weight bert/encoder/layer_6/attention/self/value/bias with shape [768]
Loading TF weight bert/encoder/layer_6/attention/self/value/kernel with shape [768, 768]
Loading TF weight bert/encoder/layer_6/intermediate/dense/bias with shape [3072]
Loading TF weight bert/encoder/layer_6/intermediate/dense/kernel with shape [768, 3072]
Loading TF weight bert/encoder/layer_6/output/LayerNorm/beta with shape [768]
Loading TF weight bert/encoder/layer_6/output/LayerNorm/gamma with shape [768]
Loading TF weight bert/encoder/layer_6/output/dense/bias with shape [768]
Loading TF weight bert/encoder/layer_6/output/dense/kernel with shape [3072, 768]
Loading TF weight bert/encoder/layer_7/attention/output/LayerNorm/beta with shape [768]
Loading TF weight bert/encoder/layer_7/attention/output/LayerNorm/gamma with shape [768]
Loading TF weight bert/encoder/layer_7/attention/output/dense/bias with shape [768]
Loading TF weight bert/encoder/layer_7/attention/output/dense/kernel with shape [768, 768]
Loading TF weight bert/encoder/layer_7/attention/self/key/bias with shape [768]
Loading TF weight bert/encoder/layer_7/attention/self/key/kernel with shape [768, 768]
Loading TF weight bert/encoder/layer_7/attention/self/query/bias with shape [768]
Loading TF weight bert/encoder/layer_7/attention/self/query/kernel with shape [768, 768]
Loading TF weight bert/encoder/layer_7/attention/self/value/bias with shape [768]
Loading TF weight bert/encoder/layer_7/attention/self/value/kernel with shape [768, 768]
Loading TF weight bert/encoder/layer_7/intermediate/dense/bias with shape [3072]
Loading TF weight bert/encoder/layer_7/intermediate/dense/kernel with shape [768, 3072]
Loading TF weight bert/encoder/layer_7/output/LayerNorm/beta with shape [768]
Loading TF weight bert/encoder/layer_7/output/LayerNorm/gamma with shape [768]
Loading TF weight bert/encoder/layer_7/output/dense/bias with shape [768]
Loading TF weight bert/encoder/layer_7/output/dense/kernel with shape [3072, 768]
Loading TF weight bert/encoder/layer_8/attention/output/LayerNorm/beta with shape [768]
Loading TF weight bert/encoder/layer_8/attention/output/LayerNorm/gamma with shape [768]
Loading TF weight bert/encoder/layer_8/attention/output/dense/bias with shape [768]
Loading TF weight bert/encoder/layer_8/attention/output/dense/kernel with shape [768, 768]
Loading TF weight bert/encoder/layer_8/attention/self/key/bias with shape [768]
Loading TF weight bert/encoder/layer_8/attention/self/key/kernel with shape [768, 768]
Loading TF weight bert/encoder/layer_8/attention/self/query/bias with shape [768]
Loading TF weight bert/encoder/layer_8/attention/self/query/kernel with shape [768, 768]
Loading TF weight bert/encoder/layer_8/attention/self/value/bias with shape [768]
Loading TF weight bert/encoder/layer_8/attention/self/value/kernel with shape [768, 768]
Loading TF weight bert/encoder/layer_8/intermediate/dense/bias with shape [3072]
Loading TF weight bert/encoder/layer_8/intermediate/dense/kernel with shape [768, 3072]
Loading TF weight bert/encoder/layer_8/output/LayerNorm/beta with shape [768]
Loading TF weight bert/encoder/layer_8/output/LayerNorm/gamma with shape [768]
Loading TF weight bert/encoder/layer_8/output/dense/bias with shape [768]
Loading TF weight bert/encoder/layer_8/output/dense/kernel with shape [3072, 768]
Loading TF weight bert/encoder/layer_9/attention/output/LayerNorm/beta with shape [768]
Loading TF weight bert/encoder/layer_9/attention/output/LayerNorm/gamma with shape [768]
Loading TF weight bert/encoder/layer_9/attention/output/dense/bias with shape [768]
Loading TF weight bert/encoder/layer_9/attention/output/dense/kernel with shape [768, 768]
Loading TF weight bert/encoder/layer_9/attention/self/key/bias with shape [768]
Loading TF weight bert/encoder/layer_9/attention/self/key/kernel with shape [768, 768]
Loading TF weight bert/encoder/layer_9/attention/self/query/bias with shape [768]
Loading TF weight bert/encoder/layer_9/attention/self/query/kernel with shape [768, 768]
Loading TF weight bert/encoder/layer_9/attention/self/value/bias with shape [768]
Loading TF weight bert/encoder/layer_9/attention/self/value/kernel with shape [768, 768]
Loading TF weight bert/encoder/layer_9/intermediate/dense/bias with shape [3072]
Loading TF weight bert/encoder/layer_9/intermediate/dense/kernel with shape [768, 3072]
Loading TF weight bert/encoder/layer_9/output/LayerNorm/beta with shape [768]
Loading TF weight bert/encoder/layer_9/output/LayerNorm/gamma with shape [768]
Loading TF weight bert/encoder/layer_9/output/dense/bias with shape [768]
Loading TF weight bert/encoder/layer_9/output/dense/kernel with shape [3072, 768]
Loading TF weight bert/pooler/dense/bias with shape [768]
Loading TF weight bert/pooler/dense/kernel with shape [768, 768]
Initialize PyTorch weight ['bert', 'embeddings', 'LayerNorm', 'beta']
Initialize PyTorch weight ['bert', 'embeddings', 'LayerNorm', 'gamma']
Initialize PyTorch weight ['bert', 'embeddings', 'position_embeddings']
Initialize PyTorch weight ['bert', 'embeddings', 'token_type_embeddings']
Initialize PyTorch weight ['bert', 'embeddings', 'word_embeddings']
Initialize PyTorch weight ['bert', 'encoder', 'layer_0', 'attention', 'output', 'LayerNorm', 'beta']
Initialize PyTorch weight ['bert', 'encoder', 'layer_0', 'attention', 'output', 'LayerNorm', 'gamma']
Initialize PyTorch weight ['bert', 'encoder', 'layer_0', 'attention', 'output', 'dense', 'bias']
Initialize PyTorch weight ['bert', 'encoder', 'layer_0', 'attention', 'output', 'dense', 'kernel']
Initialize PyTorch weight ['bert', 'encoder', 'layer_0', 'attention', 'self', 'key', 'bias']
Initialize PyTorch weight ['bert', 'encoder', 'layer_0', 'attention', 'self', 'key', 'kernel']
Initialize PyTorch weight ['bert', 'encoder', 'layer_0', 'attention', 'self', 'query', 'bias']
Initialize PyTorch weight ['bert', 'encoder', 'layer_0', 'attention', 'self', 'query', 'kernel']
Initialize PyTorch weight ['bert', 'encoder', 'layer_0', 'attention', 'self', 'value', 'bias']
Initialize PyTorch weight ['bert', 'encoder', 'layer_0', 'attention', 'self', 'value', 'kernel']
Initialize PyTorch weight ['bert', 'encoder', 'layer_0', 'intermediate', 'dense', 'bias']
Initialize PyTorch weight ['bert', 'encoder', 'layer_0', 'intermediate', 'dense', 'kernel']
Initialize PyTorch weight ['bert', 'encoder', 'layer_0', 'output', 'LayerNorm', 'beta']
Initialize PyTorch weight ['bert', 'encoder', 'layer_0', 'output', 'LayerNorm', 'gamma']
Initialize PyTorch weight ['bert', 'encoder', 'layer_0', 'output', 'dense', 'bias']
Initialize PyTorch weight ['bert', 'encoder', 'layer_0', 'output', 'dense', 'kernel']
Initialize PyTorch weight ['bert', 'encoder', 'layer_1', 'attention', 'output', 'LayerNorm', 'beta']
Initialize PyTorch weight ['bert', 'encoder', 'layer_1', 'attention', 'output', 'LayerNorm', 'gamma']
Initialize PyTorch weight ['bert', 'encoder', 'layer_1', 'attention', 'output', 'dense', 'bias']
Initialize PyTorch weight ['bert', 'encoder', 'layer_1', 'attention', 'output', 'dense', 'kernel']
Initialize PyTorch weight ['bert', 'encoder', 'layer_1', 'attention', 'self', 'key', 'bias']
Initialize PyTorch weight ['bert', 'encoder', 'layer_1', 'attention', 'self', 'key', 'kernel']
Initialize PyTorch weight ['bert', 'encoder', 'layer_1', 'attention', 'self', 'query', 'bias']
Initialize PyTorch weight ['bert', 'encoder', 'layer_1', 'attention', 'self', 'query', 'kernel']
Initialize PyTorch weight ['bert', 'encoder', 'layer_1', 'attention', 'self', 'value', 'bias']
Initialize PyTorch weight ['bert', 'encoder', 'layer_1', 'attention', 'self', 'value', 'kernel']
Initialize PyTorch weight ['bert', 'encoder', 'layer_1', 'intermediate', 'dense', 'bias']
Initialize PyTorch weight ['bert', 'encoder', 'layer_1', 'intermediate', 'dense', 'kernel']
Initialize PyTorch weight ['bert', 'encoder', 'layer_1', 'output', 'LayerNorm', 'beta']
Initialize PyTorch weight ['bert', 'encoder', 'layer_1', 'output', 'LayerNorm', 'gamma']
Initialize PyTorch weight ['bert', 'encoder', 'layer_1', 'output', 'dense', 'bias']
Initialize PyTorch weight ['bert', 'encoder', 'layer_1', 'output', 'dense', 'kernel']
Initialize PyTorch weight ['bert', 'encoder', 'layer_10', 'attention', 'output', 'LayerNorm', 'beta']
Initialize PyTorch weight ['bert', 'encoder', 'layer_10', 'attention', 'output', 'LayerNorm', 'gamma']
Initialize PyTorch weight ['bert', 'encoder', 'layer_10', 'attention', 'output', 'dense', 'bias']
Initialize PyTorch weight ['bert', 'encoder', 'layer_10', 'attention', 'output', 'dense', 'kernel']
Initialize PyTorch weight ['bert', 'encoder', 'layer_10', 'attention', 'self', 'key', 'bias']
Initialize PyTorch weight ['bert', 'encoder', 'layer_10', 'attention', 'self', 'key', 'kernel']
Initialize PyTorch weight ['bert', 'encoder', 'layer_10', 'attention', 'self', 'query', 'bias']
Initialize PyTorch weight ['bert', 'encoder', 'layer_10', 'attention', 'self', 'query', 'kernel']
Initialize PyTorch weight ['bert', 'encoder', 'layer_10', 'attention', 'self', 'value', 'bias']
Initialize PyTorch weight ['bert', 'encoder', 'layer_10', 'attention', 'self', 'value', 'kernel']
Initialize PyTorch weight ['bert', 'encoder', 'layer_10', 'intermediate', 'dense', 'bias']
Initialize PyTorch weight ['bert', 'encoder', 'layer_10', 'intermediate', 'dense', 'kernel']
Initialize PyTorch weight ['bert', 'encoder', 'layer_10', 'output', 'LayerNorm', 'beta']
Initialize PyTorch weight ['bert', 'encoder', 'layer_10', 'output', 'LayerNorm', 'gamma']
Initialize PyTorch weight ['bert', 'encoder', 'layer_10', 'output', 'dense', 'bias']
Initialize PyTorch weight ['bert', 'encoder', 'layer_10', 'output', 'dense', 'kernel']
Initialize PyTorch weight ['bert', 'encoder', 'layer_11', 'attention', 'output', 'LayerNorm', 'beta']
Initialize PyTorch weight ['bert', 'encoder', 'layer_11', 'attention', 'output', 'LayerNorm', 'gamma']
Initialize PyTorch weight ['bert', 'encoder', 'layer_11', 'attention', 'output', 'dense', 'bias']
Initialize PyTorch weight ['bert', 'encoder', 'layer_11', 'attention', 'output', 'dense', 'kernel']
Initialize PyTorch weight ['bert', 'encoder', 'layer_11', 'attention', 'self', 'key', 'bias']
Initialize PyTorch weight ['bert', 'encoder', 'layer_11', 'attention', 'self', 'key', 'kernel']
Initialize PyTorch weight ['bert', 'encoder', 'layer_11', 'attention', 'self', 'query', 'bias']
Initialize PyTorch weight ['bert', 'encoder', 'layer_11', 'attention', 'self', 'query', 'kernel']
Initialize PyTorch weight ['bert', 'encoder', 'layer_11', 'attention', 'self', 'value', 'bias']
Initialize PyTorch weight ['bert', 'encoder', 'layer_11', 'attention', 'self', 'value', 'kernel']
Initialize PyTorch weight ['bert', 'encoder', 'layer_11', 'intermediate', 'dense', 'bias']
Initialize PyTorch weight ['bert', 'encoder', 'layer_11', 'intermediate', 'dense', 'kernel']
Initialize PyTorch weight ['bert', 'encoder', 'layer_11', 'output', 'LayerNorm', 'beta']
Initialize PyTorch weight ['bert', 'encoder', 'layer_11', 'output', 'LayerNorm', 'gamma']
Initialize PyTorch weight ['bert', 'encoder', 'layer_11', 'output', 'dense', 'bias']
Initialize PyTorch weight ['bert', 'encoder', 'layer_11', 'output', 'dense', 'kernel']
Initialize PyTorch weight ['bert', 'encoder', 'layer_2', 'attention', 'output', 'LayerNorm', 'beta']
Initialize PyTorch weight ['bert', 'encoder', 'layer_2', 'attention', 'output', 'LayerNorm', 'gamma']
Initialize PyTorch weight ['bert', 'encoder', 'layer_2', 'attention', 'output', 'dense', 'bias']
Initialize PyTorch weight ['bert', 'encoder', 'layer_2', 'attention', 'output', 'dense', 'kernel']
Initialize PyTorch weight ['bert', 'encoder', 'layer_2', 'attention', 'self', 'key', 'bias']
Initialize PyTorch weight ['bert', 'encoder', 'layer_2', 'attention', 'self', 'key', 'kernel']
Initialize PyTorch weight ['bert', 'encoder', 'layer_2', 'attention', 'self', 'query', 'bias']
Initialize PyTorch weight ['bert', 'encoder', 'layer_2', 'attention', 'self', 'query', 'kernel']
Initialize PyTorch weight ['bert', 'encoder', 'layer_2', 'attention', 'self', 'value', 'bias']
Initialize PyTorch weight ['bert', 'encoder', 'layer_2', 'attention', 'self', 'value', 'kernel']
Initialize PyTorch weight ['bert', 'encoder', 'layer_2', 'intermediate', 'dense', 'bias']
Initialize PyTorch weight ['bert', 'encoder', 'layer_2', 'intermediate', 'dense', 'kernel']
Initialize PyTorch weight ['bert', 'encoder', 'layer_2', 'output', 'LayerNorm', 'beta']
Initialize PyTorch weight ['bert', 'encoder', 'layer_2', 'output', 'LayerNorm', 'gamma']
Initialize PyTorch weight ['bert', 'encoder', 'layer_2', 'output', 'dense', 'bias']
Initialize PyTorch weight ['bert', 'encoder', 'layer_2', 'output', 'dense', 'kernel']
Initialize PyTorch weight ['bert', 'encoder', 'layer_3', 'attention', 'output', 'LayerNorm', 'beta']
Initialize PyTorch weight ['bert', 'encoder', 'layer_3', 'attention', 'output', 'LayerNorm', 'gamma']
Initialize PyTorch weight ['bert', 'encoder', 'layer_3', 'attention', 'output', 'dense', 'bias']
Initialize PyTorch weight ['bert', 'encoder', 'layer_3', 'attention', 'output', 'dense', 'kernel']
Initialize PyTorch weight ['bert', 'encoder', 'layer_3', 'attention', 'self', 'key', 'bias']
Initialize PyTorch weight ['bert', 'encoder', 'layer_3', 'attention', 'self', 'key', 'kernel']
Initialize PyTorch weight ['bert', 'encoder', 'layer_3', 'attention', 'self', 'query', 'bias']
Initialize PyTorch weight ['bert', 'encoder', 'layer_3', 'attention', 'self', 'query', 'kernel']
Initialize PyTorch weight ['bert', 'encoder', 'layer_3', 'attention', 'self', 'value', 'bias']
Initialize PyTorch weight ['bert', 'encoder', 'layer_3', 'attention', 'self', 'value', 'kernel']
Initialize PyTorch weight ['bert', 'encoder', 'layer_3', 'intermediate', 'dense', 'bias']
Initialize PyTorch weight ['bert', 'encoder', 'layer_3', 'intermediate', 'dense', 'kernel']
Initialize PyTorch weight ['bert', 'encoder', 'layer_3', 'output', 'LayerNorm', 'beta']
Initialize PyTorch weight ['bert', 'encoder', 'layer_3', 'output', 'LayerNorm', 'gamma']
Initialize PyTorch weight ['bert', 'encoder', 'layer_3', 'output', 'dense', 'bias']
Initialize PyTorch weight ['bert', 'encoder', 'layer_3', 'output', 'dense', 'kernel']
Initialize PyTorch weight ['bert', 'encoder', 'layer_4', 'attention', 'output', 'LayerNorm', 'beta']
Initialize PyTorch weight ['bert', 'encoder', 'layer_4', 'attention', 'output', 'LayerNorm', 'gamma']
Initialize PyTorch weight ['bert', 'encoder', 'layer_4', 'attention', 'output', 'dense', 'bias']
Initialize PyTorch weight ['bert', 'encoder', 'layer_4', 'attention', 'output', 'dense', 'kernel']
Initialize PyTorch weight ['bert', 'encoder', 'layer_4', 'attention', 'self', 'key', 'bias']
Initialize PyTorch weight ['bert', 'encoder', 'layer_4', 'attention', 'self', 'key', 'kernel']
Initialize PyTorch weight ['bert', 'encoder', 'layer_4', 'attention', 'self', 'query', 'bias']
Initialize PyTorch weight ['bert', 'encoder', 'layer_4', 'attention', 'self', 'query', 'kernel']
Initialize PyTorch weight ['bert', 'encoder', 'layer_4', 'attention', 'self', 'value', 'bias']
Initialize PyTorch weight ['bert', 'encoder', 'layer_4', 'attention', 'self', 'value', 'kernel']
Initialize PyTorch weight ['bert', 'encoder', 'layer_4', 'intermediate', 'dense', 'bias']
Initialize PyTorch weight ['bert', 'encoder', 'layer_4', 'intermediate', 'dense', 'kernel']
Initialize PyTorch weight ['bert', 'encoder', 'layer_4', 'output', 'LayerNorm', 'beta']
Initialize PyTorch weight ['bert', 'encoder', 'layer_4', 'output', 'LayerNorm', 'gamma']
Initialize PyTorch weight ['bert', 'encoder', 'layer_4', 'output', 'dense', 'bias']
Initialize PyTorch weight ['bert', 'encoder', 'layer_4', 'output', 'dense', 'kernel']
Initialize PyTorch weight ['bert', 'encoder', 'layer_5', 'attention', 'output', 'LayerNorm', 'beta']
Initialize PyTorch weight ['bert', 'encoder', 'layer_5', 'attention', 'output', 'LayerNorm', 'gamma']
Initialize PyTorch weight ['bert', 'encoder', 'layer_5', 'attention', 'output', 'dense', 'bias']
Initialize PyTorch weight ['bert', 'encoder', 'layer_5', 'attention', 'output', 'dense', 'kernel']
Initialize PyTorch weight ['bert', 'encoder', 'layer_5', 'attention', 'self', 'key', 'bias']
Initialize PyTorch weight ['bert', 'encoder', 'layer_5', 'attention', 'self', 'key', 'kernel']
Initialize PyTorch weight ['bert', 'encoder', 'layer_5', 'attention', 'self', 'query', 'bias']
Initialize PyTorch weight ['bert', 'encoder', 'layer_5', 'attention', 'self', 'query', 'kernel']
Initialize PyTorch weight ['bert', 'encoder', 'layer_5', 'attention', 'self', 'value', 'bias']
Initialize PyTorch weight ['bert', 'encoder', 'layer_5', 'attention', 'self', 'value', 'kernel']
Initialize PyTorch weight ['bert', 'encoder', 'layer_5', 'intermediate', 'dense', 'bias']
Initialize PyTorch weight ['bert', 'encoder', 'layer_5', 'intermediate', 'dense', 'kernel']
Initialize PyTorch weight ['bert', 'encoder', 'layer_5', 'output', 'LayerNorm', 'beta']
Initialize PyTorch weight ['bert', 'encoder', 'layer_5', 'output', 'LayerNorm', 'gamma']
Initialize PyTorch weight ['bert', 'encoder', 'layer_5', 'output', 'dense', 'bias']
Initialize PyTorch weight ['bert', 'encoder', 'layer_5', 'output', 'dense', 'kernel']
Initialize PyTorch weight ['bert', 'encoder', 'layer_6', 'attention', 'output', 'LayerNorm', 'beta']
Initialize PyTorch weight ['bert', 'encoder', 'layer_6', 'attention', 'output', 'LayerNorm', 'gamma']
Initialize PyTorch weight ['bert', 'encoder', 'layer_6', 'attention', 'output', 'dense', 'bias']
Initialize PyTorch weight ['bert', 'encoder', 'layer_6', 'attention', 'output', 'dense', 'kernel']
Initialize PyTorch weight ['bert', 'encoder', 'layer_6', 'attention', 'self', 'key', 'bias']
Initialize PyTorch weight ['bert', 'encoder', 'layer_6', 'attention', 'self', 'key', 'kernel']
Initialize PyTorch weight ['bert', 'encoder', 'layer_6', 'attention', 'self', 'query', 'bias']
Initialize PyTorch weight ['bert', 'encoder', 'layer_6', 'attention', 'self', 'query', 'kernel']
Initialize PyTorch weight ['bert', 'encoder', 'layer_6', 'attention', 'self', 'value', 'bias']
Initialize PyTorch weight ['bert', 'encoder', 'layer_6', 'attention', 'self', 'value', 'kernel']
Initialize PyTorch weight ['bert', 'encoder', 'layer_6', 'intermediate', 'dense', 'bias']
Initialize PyTorch weight ['bert', 'encoder', 'layer_6', 'intermediate', 'dense', 'kernel']
Initialize PyTorch weight ['bert', 'encoder', 'layer_6', 'output', 'LayerNorm', 'beta']
Initialize PyTorch weight ['bert', 'encoder', 'layer_6', 'output', 'LayerNorm', 'gamma']
Initialize PyTorch weight ['bert', 'encoder', 'layer_6', 'output', 'dense', 'bias']
Initialize PyTorch weight ['bert', 'encoder', 'layer_6', 'output', 'dense', 'kernel']
Initialize PyTorch weight ['bert', 'encoder', 'layer_7', 'attention', 'output', 'LayerNorm', 'beta']
Initialize PyTorch weight ['bert', 'encoder', 'layer_7', 'attention', 'output', 'LayerNorm', 'gamma']
Initialize PyTorch weight ['bert', 'encoder', 'layer_7', 'attention', 'output', 'dense', 'bias']
Initialize PyTorch weight ['bert', 'encoder', 'layer_7', 'attention', 'output', 'dense', 'kernel']
Initialize PyTorch weight ['bert', 'encoder', 'layer_7', 'attention', 'self', 'key', 'bias']
Initialize PyTorch weight ['bert', 'encoder', 'layer_7', 'attention', 'self', 'key', 'kernel']
Initialize PyTorch weight ['bert', 'encoder', 'layer_7', 'attention', 'self', 'query', 'bias']
Initialize PyTorch weight ['bert', 'encoder', 'layer_7', 'attention', 'self', 'query', 'kernel']
Initialize PyTorch weight ['bert', 'encoder', 'layer_7', 'attention', 'self', 'value', 'bias']
Initialize PyTorch weight ['bert', 'encoder', 'layer_7', 'attention', 'self', 'value', 'kernel']
Initialize PyTorch weight ['bert', 'encoder', 'layer_7', 'intermediate', 'dense', 'bias']
Initialize PyTorch weight ['bert', 'encoder', 'layer_7', 'intermediate', 'dense', 'kernel']
Initialize PyTorch weight ['bert', 'encoder', 'layer_7', 'output', 'LayerNorm', 'beta']
Initialize PyTorch weight ['bert', 'encoder', 'layer_7', 'output', 'LayerNorm', 'gamma']
Initialize PyTorch weight ['bert', 'encoder', 'layer_7', 'output', 'dense', 'bias']
Initialize PyTorch weight ['bert', 'encoder', 'layer_7', 'output', 'dense', 'kernel']
Initialize PyTorch weight ['bert', 'encoder', 'layer_8', 'attention', 'output', 'LayerNorm', 'beta']
Initialize PyTorch weight ['bert', 'encoder', 'layer_8', 'attention', 'output', 'LayerNorm', 'gamma']
Initialize PyTorch weight ['bert', 'encoder', 'layer_8', 'attention', 'output', 'dense', 'bias']
Initialize PyTorch weight ['bert', 'encoder', 'layer_8', 'attention', 'output', 'dense', 'kernel']
Initialize PyTorch weight ['bert', 'encoder', 'layer_8', 'attention', 'self', 'key', 'bias']
Initialize PyTorch weight ['bert', 'encoder', 'layer_8', 'attention', 'self', 'key', 'kernel']
Initialize PyTorch weight ['bert', 'encoder', 'layer_8', 'attention', 'self', 'query', 'bias']
Initialize PyTorch weight ['bert', 'encoder', 'layer_8', 'attention', 'self', 'query', 'kernel']
Initialize PyTorch weight ['bert', 'encoder', 'layer_8', 'attention', 'self', 'value', 'bias']
Initialize PyTorch weight ['bert', 'encoder', 'layer_8', 'attention', 'self', 'value', 'kernel']
Initialize PyTorch weight ['bert', 'encoder', 'layer_8', 'intermediate', 'dense', 'bias']
Initialize PyTorch weight ['bert', 'encoder', 'layer_8', 'intermediate', 'dense', 'kernel']
Initialize PyTorch weight ['bert', 'encoder', 'layer_8', 'output', 'LayerNorm', 'beta']
Initialize PyTorch weight ['bert', 'encoder', 'layer_8', 'output', 'LayerNorm', 'gamma']
Initialize PyTorch weight ['bert', 'encoder', 'layer_8', 'output', 'dense', 'bias']
Initialize PyTorch weight ['bert', 'encoder', 'layer_8', 'output', 'dense', 'kernel']
Initialize PyTorch weight ['bert', 'encoder', 'layer_9', 'attention', 'output', 'LayerNorm', 'beta']
Initialize PyTorch weight ['bert', 'encoder', 'layer_9', 'attention', 'output', 'LayerNorm', 'gamma']
Initialize PyTorch weight ['bert', 'encoder', 'layer_9', 'attention', 'output', 'dense', 'bias']
Initialize PyTorch weight ['bert', 'encoder', 'layer_9', 'attention', 'output', 'dense', 'kernel']
Initialize PyTorch weight ['bert', 'encoder', 'layer_9', 'attention', 'self', 'key', 'bias']
Initialize PyTorch weight ['bert', 'encoder', 'layer_9', 'attention', 'self', 'key', 'kernel']
Initialize PyTorch weight ['bert', 'encoder', 'layer_9', 'attention', 'self', 'query', 'bias']
Initialize PyTorch weight ['bert', 'encoder', 'layer_9', 'attention', 'self', 'query', 'kernel']
Initialize PyTorch weight ['bert', 'encoder', 'layer_9', 'attention', 'self', 'value', 'bias']
Initialize PyTorch weight ['bert', 'encoder', 'layer_9', 'attention', 'self', 'value', 'kernel']
Initialize PyTorch weight ['bert', 'encoder', 'layer_9', 'intermediate', 'dense', 'bias']
Initialize PyTorch weight ['bert', 'encoder', 'layer_9', 'intermediate', 'dense', 'kernel']
Initialize PyTorch weight ['bert', 'encoder', 'layer_9', 'output', 'LayerNorm', 'beta']
Initialize PyTorch weight ['bert', 'encoder', 'layer_9', 'output', 'LayerNorm', 'gamma']
Initialize PyTorch weight ['bert', 'encoder', 'layer_9', 'output', 'dense', 'bias']
Initialize PyTorch weight ['bert', 'encoder', 'layer_9', 'output', 'dense', 'kernel']
Initialize PyTorch weight ['bert', 'pooler', 'dense', 'bias']
Initialize PyTorch weight ['bert', 'pooler', 'dense', 'kernel']
```

In [17]:

```python
def pandas_column_aslist(dataframe, column):
    lst = dataframe[column].tolist()
    return lst  # lista de strings

def relevances_to_pandas(dataframe, relevances):
    indexes = list(dataframe.index.values)
    labels = []
    for r in relevances:
        if r == "not relevant":
            labels.append(0)
        elif r == "relevant":
            labels.append(1)
    return pandas.Series(labels, index=indexes, name='Label')

dl_config = DLConfig(remove_punctuation="HTML")

df = pandas.read_excel("C:/Users/Zé Freitas/Desktop/Mestrado/2°Semestre/Projeto/code/datasets/dataset_final.xls
                    index_col=0)
idsfinal = pandas_column_aslist(df, "Document")
docsfinal = pmids_to_docs(idsfinal, "pg42872@alunos.uminho.pt", dl_config)
docsfinal = docsfinal[0]
print(docsfinal[0].abstract_string)
dataset_docs = docs_to_pandasdocs(docsfinal)

x_total = dataset_docs
print(x_total)

y_total = pandas.read_excel("C:/Users/Zé Freitas/Desktop/Mestrado/2°Semestre/Projeto/code/datasets/dataset_fina
                        index_col=1, usecols="B:F")

print(x_total)

relevance = pandas_column_aslist(y_total, "Relevance")
y_total = relevances_to_pandas(y_total, relevance)
print(y_total)

print(x_total["Document"][0].title_string)
print(x_total["Document"][0].abstract_string)

X_train_df, X_test_df, y_train_df, y_test_df = train_test_split(x_total, y_total, test_size=0.3, random_state=4
                                                                stratify=y_total)
print(X_train_df.shape)
print(X_test_df.shape)
print(y_train_df.shape)
print(y_test_df.shape)
```

```
Stilbenes , as important secondary metabolites of grapevine , represent central phytoalexins and therefore cons
titute an important element of basal immunity . In this study , potential genetic variation in Vitis vinifera s
sp . sylvestris , the ancestor of cultivated grapevine , was sought with respect to their output of stilbenes a
nd potential use for resistance breeding . Considerable variation in stilbene inducibility was identified in V.
vinifera ssp . sylvestris . Genotypic differences in abundance and profiles of stilbenes that are induced in re
sponse to a UV-C pulse are shown . Two clusters of stilbene 'chemovars ' emerged : one cluster showed quick and
strong accumulation of stilbenes , almost exclusively in the form of non-glycosylated resveratrol and viniferin
, while the second cluster accumulated fewer stilbenes and relatively high proportions of piceatannol and the g
lycosylated piceid . For all 86 genotypes , a time dependence of the stilbene pattern was observed : piceid , r
esveratrol , and piceatannol accumulated earlier , whereas the viniferins were found later . It was further obs
erved that the genotypic differences in stilbene accumulation were preceded by differential accumulation of the
transcripts for chalcone synthase ( CHS ) and stilbene-related genes : phenylalanine ammonium lyase ( PAL ) , s
tilbene synthase ( StSy ) , and resveratrol synthase ( RS ) . A screen of the population with respect to suscep
tibility to downy mildew of grapevine ( Plasmopara viticola ) revealed considerable variability . The subpopula
tion of genotypes with high stilbene inducibility was significantly less susceptible as compared with low-stilb
ene genotypes , and for representative genotypes it could be shown that the inducibility of stilbene synthase b
y UV correlated with the inducibility by the pathogen .
                                                   Document
25873669  <data_structures.document.Document object at 0...
24269870  <data_structures.document.Document object at 0...
17655245  <data_structures.document.Document object at 0...
16663649  <data_structures.document.Document object at 0...
33907682  <data_structures.document.Document object at 0...
...                                                     ...
32079399  <data_structures.document.Document object at 0...
26976595  <data_structures.document.Document object at 0...
20138774  <data_structures.document.Document object at 0...
27936616  <data_structures.document.Document object at 0...
28009095  <data_structures.document.Document object at 0...

[606 rows x 1 columns]
                                                   Document
25873669  <data_structures.document.Document object at 0...
24269870  <data_structures.document.Document object at 0...
17655245  <data_structures.document.Document object at 0...
16663649  <data_structures.document.Document object at 0...
33907682  <data_structures.document.Document object at 0...
...                                                     ...
32079399  <data_structures.document.Document object at 0...
26976595  <data_structures.document.Document object at 0...
20138774  <data_structures.document.Document object at 0...
27936616  <data_structures.document.Document object at 0...
28009095  <data_structures.document.Document object at 0...

[606 rows x 1 columns]
25873669    0
24269870    0
17655245    1
16663649    1
33907682    0
           ..
32079399    1
26976595    0
20138774    1
27936616    1
28009095    1
Name: Label, Length: 606, dtype: int64
Genetic diversity of stilbene metabolism in Vitis sylvestris .
Stilbenes , as important secondary metabolites of grapevine , represent central phytoalexins and therefore cons
titute an important element of basal immunity . In this study , potential genetic variation in Vitis vinifera s
sp . sylvestris , the ancestor of cultivated grapevine , was sought with respect to their output of stilbenes a
nd potential use for resistance breeding . Considerable variation in stilbene inducibility was identified in V.
vinifera ssp . sylvestris . Genotypic differences in abundance and profiles of stilbenes that are induced in re
sponse to a UV-C pulse are shown . Two clusters of stilbene 'chemovars ' emerged : one cluster showed quick and
strong accumulation of stilbenes , almost exclusively in the form of non-glycosylated resveratrol and viniferin
, while the second cluster accumulated fewer stilbenes and relatively high proportions of piceatannol and the g
lycosylated piceid . For all 86 genotypes , a time dependence of the stilbene pattern was observed : piceid , r
esveratrol , and piceatannol accumulated earlier , whereas the viniferins were found later . It was further obs
erved that the genotypic differences in stilbene accumulation were preceded by differential accumulation of the
transcripts for chalcone synthase ( CHS ) and stilbene-related genes : phenylalanine ammonium lyase ( PAL ) , s
tilbene synthase ( StSy ) , and resveratrol synthase ( RS ) . A screen of the population with respect to suscep
tibility to downy mildew of grapevine ( Plasmopara viticola ) revealed considerable variability . The subpopula
tion of genotypes with high stilbene inducibility was significantly less susceptible as compared with low-stilb
ene genotypes , and for representative genotypes it could be shown that the inducibility of stilbene synthase b
y UV correlated with the inducibility by the pathogen .
(424, 1)
(182, 1)
(424,)
(182,)
```

In [18]:

```python
dl_config = DLConfig(model_name=model_name, seed_value=seed_value)

dl_config.stop_words = None
dl_config.lower = False                #####
dl_config.remove_punctuation = False
dl_config.split_by_hyphen = False
dl_config.lemmatization = False           #####
dl_config.stems = False

#Parameters
dl_config.padding = 'post'
dl_config.truncating = 'post'

dl_config.epochs = 2     # recommended number of epochs: 2, 3, 4
dl_config.batch_size = 16    # recommended batch-size: 16 or 32  # 8, 16, 32, 64, 128
dl_config.learning_rate =  3e-5# recommended learning rate for Adam: 5e-5, 3e-5, 2e-5   # 3e-4, 1e-4,

dl_config.max_sent_len = 512      #sentences will have a maximum of "max_sent_len" words
dl_config.nmr_sentences = 1   #[1 or 2]

dl_config.validation_percentage = 30
```

In [19]:

```python
dl_config.keras_callbacks = True

if dl_config.keras_callbacks:
    checkpoint_path = str(dl_config.model_id_path) + "/model.{epoch:02d}.h5"
    keras_callbacks = [
            ModelCheckpoint(checkpoint_path,
                            verbose=0,
                            save_best_only=False,
                            save_weights_only=True,
                           save_freq='epoch')
    ]

else:
    keras_callbacks=None
```

In [20]:

```python
dl_config.tokenizer = BertTokenizer.from_pretrained('biobert_v1.1_pubmed', do_lower_case=False)

x_train, y_train, x_val, y_val =  Bert_preprocessing(X_train_df, y_train_df,
                                                     dl_config,
                                                     nmr_sentences = dl_config.nmr_sentences,
                                                     validation_percentage = dl_config.validation_percentage,
                                                     seed_value=dl_config.seed_value)
```

```
Training set with 297 samples
Validation set with 127 samples
```

In [ ]:

```python
biobert_path = './biobert_v1.1_pubmed'
if multiple_gpus:
    with strategy.scope():
        #model = Bert_FT(dl_config, learning_rate=dl_config.learning_rate, bert_name_or_path=biobert_path, bert
        model = Bert_LSTM(dl_config, learning_rate=dl_config.learning_rate,static_bert=False, bert_name_or_path

else:
    #model = Bert_FT(dl_config, learning_rate=dl_config.learning_rate, bert_name_or_path=biobert_path, bert_co
    model = Bert_LSTM(dl_config, learning_rate=dl_config.learning_rate,static_bert=False, bert_name_or_path=bio

history = model.fit(x_train, y_train,
                    epochs=dl_config.epochs,
                    batch_size=dl_config.batch_size,
                    validation_data=(x_val, y_val),
                    callbacks=keras_callbacks)
```

```
pulation', 'subs', 'subsequent', 'subsequently', 'subsp', 'subspace', 'substance', 'substances', 'substantial',
 'substantially', 'substantiate', 'substituents', 'substitute', 'substituted', 'substitution', 'substitutions',
 'substrate', 'substrates', 'subtilis', 'subtle', 'subtypes', 'subunit', 'subunits', 'success', 'successful',
 'successfully', 'sucrose', 'sue', 'suffer', 'suffering', 'sufficient', 'sufficiently', 'suffruticosa', 'suffru
ticosol', 'suffruticosols', 'sugar', 'sugars', 'suggest', 'suggested', 'suggesting', 'suggests', 'suitability',
 'suitable', 'sul', 'sulfate', 'sulfates', 'sulfation', 'sulfonic', 'sulfophenyl', 'sulforaphane', 'sulfotransf
erase', 'sulfotransferases', 'sulfur', 'sulphated', 'sulphonic', 'sulphur', 'sum', 'summarize', 'summarized',
 'summarizes', 'summarizing', 'summary', 'summer', 'sup', 'super', 'supercritical', 'superestrogenic', 'superfa
mily', 'superficial', 'superimposed', 'superior', 'supernatant', 'superoxide', 'supplement', 'supplementation',
 'supplemented', 'supplements', 'supply', 'support', 'supported', 'supporting', 'supports', 'suppose', 'suppres
s', 'suppressed', 'suppresses', 'suppressing', 'suppression', 'suppressions', 'suppressive', 'suppressor', 'sup
ramolecular', 'surface', 'surgically', 'surmise', 'surplus', 'surprisingly', 'surrounding', 'surroundings', 'su
rvey', 'surveyed', 'survival', 'susceptibility', 'susceptible', 'suspended', 'suspension', 'suspensions', 'sust
ainable', 'sustainably', 'sustained', 'sv40', 'svcam', 'swarming', 'sweet', 'sweeteners', 'sweetpotato', 'swell
ing', 'swine', 'swiss', 'switch', 'sword', 'sy5y', 'syk', 'sylvestris', 'symbionts', 'symptomatic', 'symptoms',
 'syn', 'syndrome', 'synechocystis', 'synergistic', 'synergistically', 'synergy', 'synoviocytes', 'synovium',
 'synthase', 'synthases', 'synthesis', 'synthesise', 'synthesised', 'synthesize', 'synthesized', 'synthesizes',
 'synthesizing', 'synthetase', 'synthetic', 'synthetize', 'synthetized', 'syrah', 'syringae', 'syringetin', 'sy
rups', 'system', 'systematic', 'systematically', 'systemic', 'systems', 'são', 'sémillon', 't0', 't1', 't10',
 't15', 't197a', 't3', 't47d', 't5', 't877a', 'ta100', 'tabacum', 'table', 'tablet', 'tachycardia', 'tackle',
 'tag', 'tagged', 'tailor', 'tailoring', 'tainan', 'take', 'taken', 'takes', 'taking', 'tal', 'talk', 'talks',
 'tamarixetin', 'tandem', 'tannin', 'tannins', 'tap', 'tapeworm', 'target', 'targeted', 'targeting', 'targets',
 'task', 'tatar', 'tataricum', 'tatariside', 'tau', 'tautomeric', 'taxa', 'taxifolin', 'taxol', 'taxonomic', 't
bars', 'tbilghvino', 'tc', 'tc38971', 'tcm', 'te', 'tea', 'teac', 'technical', 'technics', 'technique', 'techni
ques', 'technological', 'technologies', 'technology', 'tegument', 'telangiectasia', 'tem', 'temperate', 'temper
ature', 'temperatures', 'template', 'templates', 'temporal', 'tempting', 'ten', 'tended', 'tendrils', 'tends',
 'tensile', 'tentative', 'tentatively', 'term', 'termed', 'terminal', 'terminus', 'terms', 'teroldego', 'terpen
e', 'terpenoids', 'terrestrial', 'territory', 'tert', 'tertiary', 'test', 'tested', 'testes', 'testicular', 'te
sting', 'testosterone', 'tests', 'tethered', 'tetra', 'tetrachlorodibenzo', 'tetracycline', 'tetradecanoylphorb
ol', 'tetraethylammonium', 'tetrahydroxy', 'tetrahydroxychalcone', 'tetrahydroxyflavone', 'tetrahydroxylated',
 'tetrahydroxystilbene', 'tetrahydroxystilbenes', 'tetraketide', 'tetraketides', 'tetramer', 'tetrameric', 'tet
ramers', 'tetramethoxy', 'tetramethoxystilbene', 'tetramethyl', 'tetratricopeptide', 'tetrazolium', 'tettnang',
 'text', 'tf', 'tfs', 'tgf', 'thaliana', 'thanks', 'thaumatin', 'thawed', 'thc', 'themed', 'theobromae', 'theor
etical', 'theories', 'theory', 'therapeutic', 'therapeutical', 'therapeutics', 'therapies', 'therapy', 'theread
ter', 'thereby', 'therefore', 'thermally', 'thermodynamic', 'thermodynamics', 'thermotolerance', 'thiadiazole',
 'thickness', 'thienyl', 'thiobarbituric', 'thioester', 'thioesters', 'thiogalactoside', 'thioglycollate', 'thi
ol', 'thiols', 'third', 'thirds', 'thirteen', 'thirty', 'thompson', 'thoracic', 'thoroughly', 'though', 'though
t', 'thousandfold', 'thp', 'thr', 'threat', 'threatening', 'threats', 'three', 'threefold', 'threonine', 'thres
hold', 'thrombin', 'thrombininduced', 'thromboxane', 'throughout', 'ths', 'thsg', 'thunalbene', 'thunbergii',
 'thus', 'thymic', 'thymus', 'thyrsiflorus', 'tic', 'tidylinositol', 'ties', 'tifton', 'tify', 'tight', 'tightl
y', 'tigr', 'tillers', 'time', 'times', 'timing', 'tintorera', 'tip', 'tips', 'tissue', 'tissues', 'titer', 'ti
ters', 'tlc', 'tms', 'tnbc', 'tnf', 'tobacco', 'tocopherol', 'tocopherols', 'tocsy', 'today', 'tof', 'tofms',
 'together', 'tolerability', 'tolerance', 'tolerant', 'toll', 'toluene', 'tomato', 'tomatoes', 'tomentosa', 'to
mloxb', 'tomography', 'tonality', 'tonghua', 'tonic', 'took', 'tool', 'tools', 'top', 'topic', 'topical', 'topi
cally', 'topoisomerase', 'topology', 'torachrysone', 'torulene', 'total', 'totally', 'toughness', 'toward', 'to
wards', 'toxic', 'toxicities', 'toxicity', 'toxicological', 'toxicologists', 'toxigenic', 'toxins', 'toyopear
l', 'tpa', 'tpc', 'tph', 'tpr', 'tps', 'tr', 'trace', 'traced', 'traces', 'tracker', 'tracking', 'trade', 'trad
itional', 'traditionally', 'traductional', 'trafficking', 'trait', 'traits', 'tranported', 'trans', 'transamina
se', 'transcribed', 'transcript', 'transcriptase', 'transcription', 'transcriptional', 'transcriptionally', 'tr
anscriptions', 'transcriptome', 'transcriptomic', 'transcripts', 'transduced', 'transducers', 'transducing', 't
ransduction', 'transfected', 'transfection', 'transfer', 'transferase', 'transferases', 'transferred', 'transfe
rring', 'transform', 'transformants', 'transformation', 'transformations', 'transformed', 'transgene', 'transge
nes', 'transgenic', 'transgenics', 'transient', 'transiently', 'transit', 'transition', 'transitions', 'transla
ted', 'translation', 'translational', 'translocated', 'translocation', 'translocations', 'transmembrane', 'tran
smethylation', 'transmission', 'transpeptidase', 'transport', 'transportation', 'transported', 'transporter',
 'transporters', 'transposable', 'transposase', 'transresveratrol', 'transthyretin', 'trap', 'trapped', 'trappi
ng', 'tre', 'treat', 'treated', 'treating', 'treatment', 'treatments', 'tree', 'trees', 'tremendously', 'tremul
a', 'tremulae', 'tremulax', 'tremuloides', 'trend', 'triacetic', 'triad', 'trials', 'tribe', 'tribes', 'trichod
erma', 'trichomes', 'trifluoroacetamide', 'trifluorostilbene', 'trifolium', 'trigger', 'triggered', 'triggers',
 'trihydroxy', 'trihydroxybenzoic', 'trihydroxylated', 'trihydroxyphenanthrene', 'trihydroxystilbene', 'triketi
de', 'trimer', 'trimeric', 'trimers', 'trimethoxy', 'trimethoxystilbene', 'trimethyl', 'trimethylether', 'trime
thylsilyl', 'triphenolic', 'triphosphate', 'triple', 'triplicated', 'trismethoxyresveratrol', 'trisphosphate',
 'triterpenes', 'triterpenoids', 'tritici', 'trolox', 'true', 'trunk', 'trypan', 'ts', 'tsg', 'tso', 'ttas', 't
tr', 'tube', 'tuber', 'tuberosum', 'tubers', 'tubes', 'tubingensis', 'tubular', 'tubulin', 'tumefaciens', 'tumo
r', 'tumorigenesis', 'tumors', 'tumour', 'tumours', 'tunel', 'tuning', 'turbot', 'turkey', 'turn', 'turned', 't
urnover', 'turns', 'twelve', 'twenty', 'twice', 'twigs', 'twist', 'twist1', 'two', 'twofold', 'type', 'types',
 'typhimurium', 'typical', 'typically', 'tyr', 'tyrosinase', 'tyrosine', 'tyrosol', 'uae', 'ubiquitous', 'udp',
 'ufgt', 'ugt', 'ugts', 'uhplc', 'ultimately', 'ultra', 'ultrasonication', 'ultrasound', 'ultrastructural', 'ul
traviolet', 'ulva', 'um', 'umbelliferone', 'umbilical', 'ump', 'unable', 'unaffected', 'unaltered', 'unambiguou
s', 'unambiguously', 'unavailable', 'uncertain', 'unchanged', 'uncharacterized', 'uncinula', 'unclear', 'uncont
rolled', 'unconventional', 'undamaged', 'underexploited', 'underexplored', 'undergo', 'undergoes', 'undergoin
g', 'undergone', 'underling', 'underlying', 'underpinnings', 'underscore', 'understand', 'understanding', 'unde
rstood', 'undertaken', 'underutilized', 'underway', 'underwent', 'undescribed', 'undesirable', 'undetectable',
 'undetected', 'undulatum', 'unequivocally', 'unexpected', 'unexpectedly', 'unexplored', 'unfavourable', 'unfol
ding', 'unfriendly', 'unicellular', 'unidentified', 'uniform', 'unigenes', 'unilamellar', 'uninoculated', 'unin
tentional', 'unique', 'uniquely', 'unit', 'united', 'units', 'universal', 'university', 'unknown', 'unlike', 'u
nlimited', 'unmodified', 'unnatural', 'unpaired', 'unpeeled', 'unprecedented', 'unravel', 'unraveled', 'unravel
led', 'unreduced', 'unrelated', 'unresolved', 'unripe', 'unsaturated', 'unspecific', 'unstable', 'unstimulate
d', 'unstressed', 'unstructured', 'unsustainable', 'unsweetened', 'untargeted', 'untested', 'untouched', 'untra
nsformed', 'untreated', 'unusual', 'unusually', 'unveil', 'update', 'updated', 'uplc', 'upon', 'upper', 'upregu
late', 'upregulated', 'upregulates', 'upregulating', 'upregulation', 'upscaled', 'upstream', 'uptake', 'upuna',
 'urate', 'urea', 'urgent', 'uric', 'uricase', 'uricosuric', 'uridine', 'urinary', 'urine', 'ursolate', 'ursoli
c', 'urushiol', 'us', 'use', 'used', 'useful', 'usefulness', 'uses', 'using', 'ustilago', 'usual', 'usually',
 'uterine', 'utility', 'utilization', 'utilize', 'utilized', 'utilizes', 'utilizing', 'uv', 'uvb', 'uvc', 'v17
7', 'v196m', 'vaccenic', 'vaccinium', 'vacpk20', 'vacuole', 'vacuoles', 'vacuum', 'vad', 'val', 'val882', 'vale
nce', 'validated', 'validating', 'validation', 'validity', 'valpolicella', 'valuable', 'value', 'valued', 'valu
es', 'vanicoside', 'vanicosides', 'vanillic', 'vanillin', 'vapal1', 'vapal2', 'vapal3', 'vapal5', 'var', 'varia
bility', 'variable', 'variant', 'variants', 'variation', 'variations', 'varied', 'varies', 'varietal', 'varieti
es', 'variety', 'various', 'vary', 'varying', 'vascular', 'vasculature', 'vasorelaxant', 'vasorelaxing', 'vasos
pastic', 'vast', 'vasts', 'vasts1', 'vasts10', 'vasts19', 'vasts2', 'vasts3', 'vasts4', 'vasts5', 'vasts6', 'va
sts7', 'vasts8', 'vasts9', 'vastss', 'vateria', 'vatica', 'vaticanol', 'vaticanols', 'vatp16', 'vector', 'vecto
rs', 'vegetable', 'vegetables', 'vegetative', 'vein', 'veins', 'veltliner', 'venezuelae', 'venous', 'vent', 've
ntricular', 'venturia', 'venues', 'vera', 'veraison', 'verapamil', 'veratrum', 'verified', 'verifing', 'verif
y', 'versa', 'versatile', 'versatility', 'version', 'versus', 'vertebrates', 'vertical', 'vessel', 'vessels',
 'vf', 'via', 'viability', 'viable', 'vice', 'view', 'viewed', 'viewpoint', 'vigorous', 'vii', 'vimentin', 'vi
n', 'vinalco', 'vine', 'vineatrol', 'vines', 'vineyard', 'vineyards', 'vinhão', 'vinifera', 'viniferap', 'vinif
erin', 'viniferinas', 'viniferinglucoside', 'viniferins', 'vinification', 'viniverin', 'vintage', 'vintages',
 'violacein', 'violaceum', 'violent', 'violet', 'viral', 'virally', 'virgaurea', 'viride', 'virtually', 'virtu
e', 'virulence', 'virus', 'visible', 'visionht', 'visit', 'visualized', 'vit', 'vitaceae', 'vital', 'vitamin',
 'vitamins', 'viticola', 'viticultural', 'viticulture', 'vitis', 'vitis3', 'vitisamurensis', 'vitisifuran', 'vi
tisin', 'vitisinol', 'vitisvinifera', 'vitro', 'vitus', 'vivo', 'vla', 'vlapx6', 'vldl', 'vlgpx5', 'vlmyb14',
 'vlprx13', 'vlprx21', 'vlprx35', 'vlprx40', 'vlprx50', 'vlprxs', 'vlrsgt', 'vmax', 'volatile', 'voltage', 'vol
ume', 'volunteers', 'vpglo1', 'vpromt', 'vps', 'vpsts', 'vpsts29', 'vpsts36', 'vqbzip1', 'vqmapkkk38', 'vqsnrk
2', 'vqsts', 'vqsts21', 'vqsts36', 'vqsts6', 'vranec', 'vrromt', 'vrromtsyn', 'vrsts', 'vs', 'vst', 'vst1', 'vs
t2', 'vsti', 'vt', 'vvabcg44', 'vvabf2', 'vvimyb13', 'vvimyb14', 'vvimyb15', 'vvists', 'vvists16', 'vvists29',
 'vvists46', 'vvists6', 'vvists7', 'vvistss', 'vviwrky03', 'vviwrky24', 'vviwrky43', 'vviwrky53', 'vvmyb14', 'v
vnpr1', 'vvromt', 'vvsts', 'vvsts1', 'vvsts15', 'vvsts2', 'vvwrky18', 'vvwrky8', 'váh', 'véraison', 'w20a', 'wa
dsworth', 'waf1', 'wall', 'walls', 'wang', 'wanted', 'wants', 'ward', 'warrant', 'warranted', 'waste', 'waste
d', 'water', 'wavelength', 'way', 'ways', 'wb', 'weak', 'weaker', 'weaned', 'weather', 'web', 'weed', 'week',
 'weeks', 'weight', 'weights', 'well', 'western', 'whatever', 'wheat', 'whereas', 'whereby', 'whether', 'whit
e', 'whole', 'wholesomeness', 'whose', 'wi38', 'wi38va', 'wide', 'widely', 'widen', 'wider', 'widespread', 'wid
th', 'wild', 'wiley', 'willamette', 'wilsonol', 'wilt', 'wilting', 'wine', 'winegrape', 'winemaking', 'winery',
 'wines', 'winter', 'wistar', 'withanolides', 'withering', 'within', 'without', 'wittig', 'wk', 'wl', 'women',
 'wonder', 'wondrous', 'wood', 'woods', 'woody', 'work', 'worked', 'workers', 'works', 'world', 'worldwide', 'w
orms', 'worse', 'wort', 'worth', 'wos', 'would', 'wound', 'wounding', 'writhings', 'wrky', 'ws', 'wt', 'xada',
 'xanthine', 'xbohemica', 'xenobiotic', 'xenograft', 'xenopus', 'xhfd', 'xhoi', 'xiangfei', 'xinjiang', 'xl',
 'xo', 'xtt', 'xuhua', 'xylella', 'xylem', 'y1h', 'y2h', 'yarrow', 'yarrowia', 'yd', 'year', 'years', 'yeast',
 'yeasts', 'yellow', 'yellowish', 'yet', 'yield', 'yielded', 'yielding', 'yields', 'yl', 'yolk', 'young', 'youn
ger', 'youngest', 'yucca', 'yuccaols', 'z168', 'zalcitabine', 'zealand', 'zebrafish', 'zero', 'zerumbone', 'zha
ng', 'zhonghua', 'zicui', 'zidovudine', 'zileuton', 'zim', 'zinnia', 'zipper', 'zn', 'zone', 'zones', 'zoospor
e', 'zoospores', 'zr', 'zucc', 'zweigeltrebe', 'zymography', 'µg', 'µm', 'ß3', 'ģellewża', 'κb', 'αvß3', 'αiβγ2
s', 'αiβ2γ2s', 'αiβ1γ2s', 'α2β2γ2s', 'α3β2γ2s', 'α4β2γ2s', 'α5β2γ2s', 'β1', 'δcao', 'δe', 'κb', 'μg', 'μgml', 'μm', 'μmo
l', 'тhe']
9403
['stilbenes', 'important', 'secondary', 'metabolites', 'grapevine', 'represent', 'central', 'phytoalexins', 'th
erefore', 'constitute', 'important', 'element', 'basal', 'immunity', 'study', 'potential', 'genetic', 'variatio
n', 'vitis', 'vinifera', 'ssp', 'sylvestris', 'ancestor', 'cultivated', 'grapevine', 'sought', 'respect', 'outp
ut', 'stilbenes', 'potential', 'use', 'resistance', 'breeding', 'considerable', 'variation', 'stilbene', 'induc
ibility', 'identified', 'vinifera', 'ssp', 'sylvestris', 'genotypic', 'differences', 'abundance', 'profiles',
 'stilbenes', 'induced', 'response', 'uv', 'pulse', 'shown', 'two', 'clusters', 'stilbene', 'chemovars', 'emerg
ed', 'one', 'cluster', 'showed', 'quick', 'strong', 'accumulation', 'stilbenes', 'almost', 'exclusively', 'for
m', 'non', 'glycosylated', 'resveratrol', 'viniferin', 'second', 'cluster', 'accumulated', 'fewer', 'stilbene
s', 'relatively', 'high', 'proportions', 'piceatannol', 'glycosylated', 'piceid', '86', 'genotypes', 'time', 'd
ependence', 'stilbene', 'pattern', 'observed', 'piceid', 'resveratrol', 'piceatannol', 'accumulated', 'earlie
r', 'whereas', 'viniferins', 'found', 'later', 'observed', 'genotypic', 'differences', 'stilbene', 'accumulatio
n', 'preceded', 'differential', 'accumulation', 'transcripts', 'chalcone', 'synthase', 'chs', 'stilbene', 'rela
ted', 'genes', 'phenylalanine', 'ammonium', 'lyase', 'pal', 'stilbene', 'synthase', 'stsy', 'resveratrol', 'syn
thase', 'rs', 'screen', 'population', 'respect', 'susceptibility', 'downy', 'mildew', 'grapevine', 'plasmopar
a', 'viticola', 'revealed', 'considerable', 'variability', 'subpopulation', 'genotypes', 'high', 'stilbene', 'i
nducibility', 'significantly', 'less', 'susceptible', 'compared', 'low', 'stilbene', 'genotypes', 'representati
ve', 'genotypes', 'could', 'shown', 'inducibility', 'stilbene', 'synthase', 'uv', 'correlated', 'inducibility',
 'pathogen']
```


In [3]:

```python
y_total = pd.read_excel("C:/Users/Zé Freitas/Desktop/Mestrado/2°Semestre/Projeto/code/datasets/dataset_final.xl
                        index_col=1, usecols="B:F")

relevance = pandas_column_aslist(y_total, "Relevance")
y_total = relevances_to_pandas(y_total, relevance)
print(y_total)

# as variáveis de entrada serão os scores das features (matriz de tfidf)
X_train_df, X_test_df, y_train_df, y_test_df = train_test_split(X, y_total, test_size=0.3, random_state=42,
                                                                stratify=y_total)
print(X_train_df.shape)
print(X_test_df.shape)
print(y_train_df.shape)
print(y_test_df.shape)
```

```
25873669    0
24269870    0
17655245    1
16663649    1
33907682    0
           ..
32079399    1
26976595    0
20138774    1
27936616    1
28009095    1
Name: Label, Length: 606, dtype: int64
(424, 9403)
(182, 9403)
(424,)
(182,)
```

Os **DataFrames de treino** criados contêm, nas linhas, os documentos, e nas colunas todas as features geradas anteriormente pelo processo de **TF-IDF**. Este DataFrame é muito desproporcional em termos de número de linhas e número de colunas, pelo que a melhor opção será reduzir o número de colunas de modo a evitar o over-fiting do modelo (meter descrição).
Tendo isso em contra, foi utilizado o package **RFE** do sklearn, que permite realizar feature selection de acordo com os melhores scores da matriz gerada a cima.

In [4]:

```python
estimator = SVC(kernel= "linear")
selector = RFE(estimator, n_features_to_select=300, step=1)
selector_norm = selector.fit(X_train_df, y_train_df)
```

A pipeline da célula anterior permite realizar, em primero lugar, uma standardização dos dados da matriz de frequências de termos por documento, e dizer ao mesmo tempo qual o modelo SVM que vai ser utilizado.
Ao passar este estimador para a função RFE, podemos escolher o número de features que queremos que sejam selecionadas, neste caso, como este dataset tem 606 linhas (Documentos) ficou decidido selecionar apenas as 300 melhores features.

A seguir passamos inevitavelmente para a predição da relevância dos documentos presentes no test_set.

In [5]:

```python
pred = selector_norm.predict(X_test_df)
selector_norm.score(X_test_df, y_test_df)
```

Out[5]: 0.8131868131868132

In [9]:

```python
cm = confusion_matrix(y_test_df.values, pred)
cm = np.flip(cm)
print(cm)

acc = accuracy_score(y_test_df, pred)
print(acc)  # é igual ao método .score()

prec = precision_score(y_test_df, pred)
print(prec)

rec = recall_score(y_test_df, pred)
print(rec)

f1 = f1_score(y_test_df, pred)
print(f1)
```

```
[[ 34  25]
 [  9 114]]
0.8131868131868132
0.7906976744186046
0.576271186440678
0.6666666666666667
```

In [ ]:

```python
# Cross Validation
scores = cross_validate(selector_norm, X, y_total, cv=5, scoring=("accuracy", "precision", "recall", "f1"))

print(scores["test_accuracy"])
print(scores["test_precision"])
print(scores["test_recall"])
print(scores["test_f1"])
```

Para os dados standardizados:

In [10]:

```python
scaler = StandardScaler(with_mean=False)  # nao pode usar a média por ser sparse matrix
scaler.fit(X)
X_scaled = scaler.transform(X)
print(X_scaled)  # matriz de TF-IDFs strandardizada

# Divisão em train e test
X_train_df_scaled, X_test_df_scaled, y_train_df, y_test_df = train_test_split(X_scaled, y_total, test_size=0.3,
                                                                              random_state=42, stratify=y_total
print(X_train_df_scaled.shape)
print(X_test_df_scaled.shape)
print(y_train_df.shape)
print(y_test_df.shape)

selector = RFE(estimator, n_features_to_select=300, step=1)
selector_scaled = selector.fit(X_train_df_scaled, y_train_df)
```

```
  (0, 6290)	2.0855101269787246
  (0, 2294)	5.046915214520747
  (0, 2320)	2.540524587697452
  (0, 7396)	6.593205052025273
  (0, 5112)	2.079495968996465
  (0, 2071)	2.2275908813912486
  (0, 8360)	4.347418808530979
  (0, 4985)	3.3052758971804423
  (0, 7885)	2.1164774459151747
  (0, 8257)	24.637403533906777
  (0, 9030)	14.159581400419084
  (0, 7501)	2.5752497541527446
  (0, 9148)	2.2683605901527204
  (0, 6624)	4.278739955210288
  (0, 5492)	2.9271603949145586
  (0, 3009)	4.678351574202174
  (0, 8359)	7.296720701472103
  (0, 6730)	5.548153735433048
  (0, 7731)	15.250471933293142
  (0, 7623)	1.8389176699125274
  (0, 8230)	5.695532441234388
  (0, 6216)	4.444652191346603
  (0, 5142)	3.7786302714815663
  (0, 795)	11.166037776156353
  (0, 6468)	3.5847344761535846
  :	:
  (605, 9017)	4.258386510359176
  (605, 4495)	4.385817297292112
  (605, 1771)	1.0311604843675872
  (605, 2525)	3.102026903971881
  (605, 1409)	2.362185837958699
  (605, 5949)	4.377106317082512
  (605, 8320)	3.163558391796091
  (605, 4761)	2.9302040731433734
  (605, 4252)	2.583171722773447
  (605, 5374)	2.594260241674152
  (605, 5449)	8.055180428695715
  (605, 798)	4.2489392460992175
  (605, 3836)	1.3873923751179975
  (605, 9040)	2.71090934074737
  (605, 6619)	2.496318072750146
  (605, 6616)	2.1694249126329677
  (605, 8394)	1.9304933601092886
  (605, 2105)	1.0443268448100544
  (605, 3463)	1.2473337021223327
  (605, 3414)	5.523856165683722
  (605, 3846)	1.2692056731889565
  (605, 7482)	0.4657677876467764
  (605, 8558)	4.341087337111881
  (605, 5381)	1.949860511844126
  (605, 7745)	2.5761616522242377
(424, 9403)
(182, 9403)
(424,)
(182,)
[[121   2]
 [ 42  17]]
```

In [12]:

```python
pred_scaled = selector.predict(X_test_df_scaled)
selector_scaled.score(X_test_df_scaled, y_test_df)
```

Out[12]: 0.7582417582417582

In [11]:

```python
cm = confusion_matrix(y_test_df.values, pred_scaled)
cm = np.flip(cm)
print(cm)

acc = accuracy_score(y_test_df, pred_scaled)
print(acc)  # é igual ao método .score()

prec = precision_score(y_test_df, pred_scaled)
print(prec)

rec = recall_score(y_test_df, pred_scaled)
print(rec)

f1 = f1_score(y_test_df, pred_scaled)
print(f1)
```

```
[[ 17  42]
 [  2 121]]
0.7582417582417582
0.8947368421052632
0.288135593220339
0.43589743589743596
```

In [14]:

```python
# Cross Validation
# scores = cross_validate(selector_norm, X, y_total, cv=5, scoring=("accuracy", "precision", "recall", "f1"))
print("Cross validation metrics scores for each fold")
print("Accuracy scores:", scores["test_accuracy"])
print("Precision scores:",scores["test_precision"])
print("Recall scores:",scores["test_recall"])
print("f1 scores:",scores["test_f1"])

print("-----------------------------------------------------------------------")
# Cross Validation standard data
# scores_stand = cross_validate(selector_scaled, X_scaled, y_total, cv=5, scoring=("accuracy", "precision", "re
print("Cross validation metrics scores for each fold for standardized data")
print("Accuracy scores:",scores_stand["test_accuracy"])
print("Precision scores:",scores_stand["test_precision"])
print("Recall scores:",scores_stand["test_recall"])
print("F1 scores:",scores_stand["test_f1"])
```

```
Cross validation metrics scores for each fold
Accuracy scores: [0.81147541 0.81818182 0.79338843 0.80991736 0.71900826]
Precision scores: [0.78571429 0.9047619  0.85       0.9        0.55102041]
Recall scores: [0.56410256 0.48717949 0.43589744 0.46153846 0.69230769]
f1 scores: [0.65671642 0.63333333 0.57627119 0.61016949 0.61363636]
----------------------------------------------------------------------------------------------------
Cross validation metrics scores for each fold for standardized data
Accuracy scores: [0.69672131 0.73553719 0.74380165 0.7768595  0.74380165]
Precision scores: [0.625      1.         0.83333333 0.92857143 0.65384615]
Recall scores: [0.12820513 0.17948718 0.25641026 0.33333333 0.43589744]
F1 scores: [0.21276596 0.30434783 0.39215686 0.49056604 0.52307692]
```

Em modelos de SVM é, numa grande maioria das vezes, aconselhada a standardização dos dados, devido à forma como o modelo atua, e por isso, esta passo foi também incluído, usando a função *StandardScaler* do sklearn, a matriz *term-document* fruto do processo **TF-IFD**, foi standardizada segundo apenas o seu desvio padrão, uma vez que se trato de uma *Sparse Matrix*, ou seja, é uma matriz composta essencialmente por elementos nulos. Os dados standardizadas revelaram um declínio bastante acentuado no valor de score (mean accuracy), pelo que se pode concluir que não se deve proceder a standardização dos dados.

Conclusões no paper!