Treball de Fi de Grau

Grau en Enginyeria en Tecnologies Industrials

Desenvolupament d'una aplicació web per a NLP

MEMÒRIA

Autor: Max Montoliu Torruella
Director: Alexandre Perera Lluna

Convocatòria: Juliol 2020

Escola Tècnica Superior
d'Enginyeria Industrial de Barcelona

Resum

L'objectiu del treball és la implementació d'un aplicatiu web capaç de facilitar l'accés al públic general d'un mètode de traducció capaç de relacionar descripcions simptomàtiques amb termes corrents amb una ontologia clínica de referència.

S'expliquen els fonaments teòrics que fan possible la traducció automàtica i en concret, es descriuen els mètodes emprats

S'estudien les eines existents per a integrar aplicatius webs capaços d'efectuar prediccions de models d'aprenentatge profund ja entrenats amb llibreries com Keras, Per mitjà del desenvolupament de diversos models de processat de llenguatge natural i a fi de poder emetre prediccions en el navegador web amb la llibreria TensorFlow.js.

Es desenvolupa un estudi sobre les solucions actuals per a la representació gràfica d'informació complexa. En un primer terme s'opta per llibreries desenvolupades amb Python. A continuació i com a conseqüència de l'estudi de llibreries especialitzades en la representació d'ontologies, es desenvolupen solucions que resulten especialment útils per a l' entorn web gràcies a la llibreria D3.js

S'arriba a la conclusió que una proposta d'aplicatiu desenvolupada amb Node.js i Express permetrà una millor solució a la problemàtica. Assolint-se un aplicatiu basada tant en la banda del servidor com en la del client capaç de desenvolupar la tasca esmentada així com de transmetre informació complexa sobre estructures ontològiques de manera gràfica, intuïtiva i segons els interessos de l'usuari.

Sumari

1. Glossari

DAG abreviació per *Directed acíclic graph*

DL abreviació per Deep learning

DNN abreviació per Deep neural networks

GRU abreviació per Gated recurrent units

HPO abreviació per Human Phenotype Ontology

HTTP abreviació per hyperText Transfer Protocol

LSTM abreviació per Long short term memory

ML abreviació per Machine Learning

NLP abreviació per Natural Language Processing

RNN abreviació per Recurrent Neural Networks

S4r abreviació per Share4Rare

WE abreviació per Word embeding

2. Introducció

Des del naixement d' Internet als anys 70, ha esdevingut una xarxa de telecomunicacions d'accés públic capaç de connectar dispositius i terminals arreu del mon. És a partir de l'aparició del protocol HyperText Transfer Protocol (HTTP) que es facilita l'intercanvi de documents a través d'internet interpretables per mitjà dels navegadors web.

L'aparició dels navegadors web va permetre la democratització de l'ús d'internet i el seu ús i difusió segueix augmentant d'una manera increïble. Ha suposat una revolució sense precedents en la creació i transmissió d'informació arreu del planeta.

Aquesta gran quantitat d'informació juntament amb la millora dels sistemes de computació, ha permès el desenvolupament d'algoritmes amb capacitats per dur a terme tasques que fa menys de mig segle haurien sigut difícilment imaginables.

Un dels exemples més notoris, és el processament de llenguatge natural (NLP) per les seves sigles en anglès. Branca de les ciències de la computació i intel·ligència artificial que permet el tractament automàtic del llenguatge natural.

Una de les aplicacions del NLP és el de traducció. Consisteix en la interpretació d'allò que es diu en un llenguatge i en la traducció en un altre amb la voluntat de preservar la informació disponible.

En el seu moment, els navegadors web van fer accessible la informació que internet permetia difondre. Ara al seu torn el NLP permet que aquesta informació esdevingui a més, comprensible.

L'objectiu d'aquest treball, és d'establir una un pont de comunicació capaç de facilitar l'accés i comprensió d'informació d'un gran nivell de complexitat al públic general.

2.1. Motivació

La voluntat que motiva la creació d'aquest projecte és la d'utilitzar la tecnologia actual per tal de facilitar l'accés a diagnòstic pertinent a totes aquelles persones que pateixen de malalties rares.

Share4Rare (s4r, www.share4rare.org) és una iniciativa que neix amb la voluntat d'oferir una plataforma per tal de permetre l'intercanvi d'informació entre l'entorn de gent que presenta tumors rars, trastorns neuromusculars i malalties sense diagnòstic. A fi de permetre l'accés d'informació clínica de qualitat als pacients per tal de poder millorar el tractament així com el

diagnosis.

L'existència d'un projecte capaç de suplir el buit existent entre terminologies clíniques com l'Human Phenotype Ontology (HPO) i la descripció simptomàtica expressada amb termes corrents, fa pensar en la potencialitat d'implementar una plataforma web per tal de fer accessible aquest coneixement a una comunitat que necessita d'aquest coneixement,

2.2. Objectius del projecte i abast

L'objectiu central d'aquest treball és oferir un aplicatiu web capaç de facilitar l'accés al públic a un model de traducció de termes comuns a HPO. A més, pretén servir d'eina per a efectuar el procés invers i ampliar la familiarització a la terminologia clínica i la complexitat de l'estructura l'HPO a través d'eines de visualització.

Per tal de complir amb l'objectiu principal,

- Comprensió d'un model existent de traducció de termes comuns a HPO
- Estudiar les possibles solucions per a implementar un aplicatiu web capaç d'executar un model de NLP .
- Dotar l'aplicatiu web de la capacitat d'enviar al model les peticions de l'usuari
- Dotar a l'aplicatiu web de la capacitat de transferir les prediccions del model a l'usuari
- Utilitzar eines per a la visualització termes HPO i fer-los accessibles al públic general dotant-los de context.

El desenvolupament del projecte i l'assoliment dels seus objectius es du a terme en el període de 8 mesos.

3. L' Human Phenotype Ontology

L' **Human Phenotype Ontology** és un projecte publicat l'any 2008 amb l'objectiu d' oferir un pont d'entesa entre la biologia d'escala genòmica i la visió centrada en la malaltia sobre la pato biologia humana per mitjà de característiques fenotípiques (manifestacions observables d'un organisme fruit de la interacció entre genotip i ambient).En concret, de les anomalies fenotípiques.

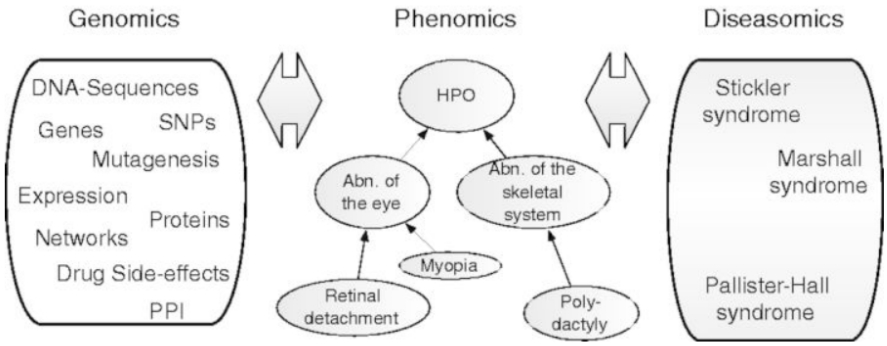

Figura 1: Representació d'HPO [1]

En les ciències de la computació,[2] una ontologia és una representació formal de coneixement per mitjà d'un conjunt de conceptes i les relacions entre aquests en un àmbit determinat. Es doncs a través d'aquesta especificació formal i explicita d'una conceptualització compartida que es configura una ontologia.

Els tres components de l'HPO són el vocabulari fenotípic, les anotacions malaltia-fenotip i els algorismes que permeten operar sobre aquests. Des del seu naixement, l'HPO ha enriquit i refinat la seva terminologia clínica en varies àrees i ha afegit més de 6000 termes nous assolint pràcticament 15000 termes en l'actualitat.

L'estructura de l'HPO s'estructura com un directed acíclic graph (DAG). En ser acílic, el graf no pot presentar relació cíclica. Però un terme pot ser descendent de més d'un terme. L' especialització en els nodes fills respecte nodes pares (menys especialitzats) s'assoleix per mitjà de la relació transitiva **is-a** de manera que les anotacions són heretades per tots els camins fins com a molt l'arrel. De tota l'ontologia Per exemple, el terme *Abnormal ventricular*

septum morphology és descendent d'Abnormal cardíac septum morphology i Abnormal cardíac ventricle morphology.

Cada terme de l'HPO descriu una anomalia clínica i es caracteritza per un id i name. També solen presentar una descripció, sinònims nodes pares (relació transitiva is-a) i referencies d'altres ontologies. Per exemple, el terme *Abnormal ventricular septum morphology:*

- **Id:** HP:0010438
- **Name**: "*Abnormal ventricular septum morphology"*
- **Definition:** "*A structural abnormality of the interventricular septum"*
- **Synonyms:** "*Abnormal interventricular septum morphology"*
- **Is a:** *d'Abnormal cardíac septum morphology i Abnormal cardíac ventricle morphology.*
- **Xref:** ORCID:0000-0001-5208-3432

L'ontologia està dividida en 5 sub-ontologies de les quals cada un dels termes n'ha de formar part:

- Phenotypic abnormality: Amb més de 14000 termes és la principal ontologia i consta de les descripcions de les anomalies clíniques
- Clinical modifier: L'ontologia conté més de 100 classes que descriuen els modificadors de símptomes clínics.
- Clinical course: Amb menys de 50 termes es descriu el curs que pren una malaltia des del seu inici fins a la resolució o mort de l'individu afectat així com la seva progressio en el temps com per exemple *Age of death*
- Mode of inheritance: Aquesta subontologia també consta de menys de 50 termes que descriuen el mode d'herència.
- Frequency: Representa la freqüència de certs fenotips.

4. Estat de l'art

4.1. Machine i deep learning:

Un algoritme de ML pot ser definit com aquell que és capaç de millorar en l'execució d'una tasca concreta a partir de l'experiència i en base a una mesura de rendiment. Si el seu rendiment en la tasca a través de la mesura millora en base a l'experiència de manera autònoma.

La tasca de traducció autònoma de termes comuns a HPO es realitzada a partir d'arquitectures d'aprenentatge profund(DL). El DL són el conjunt de tècniques de ML que es caracteritzen per l'ús de varis nivells amb funcions no lineals entrenats a partir de una gran quantitat de dades.

El bloc essencial de les arquitectures de DL és la nuerona artificial. Aquesta pren el vector d'entrada, el multiplica pels pesos i hi suma bias per subministrar-ho a la funció d'activació (no lineal).

A partir de la definició d'una mètrica d'error y la comparació entre el valor predit per la funció d'activació i el real es du a terme l'entrenament de la neurona. A fi, d'entrenar-ne' els pesos amb la voluntat de minimitzar la funció loss

A través de la connexió de varis nivells és creen les deep neural Networks (DNN).

4.2. Recurrent Neural Networks:

Les Recurrent Neural Networks (RNN), són el conjunt de xarxes neuronals que s'ocupen de la tasca de processar informació seqüencial. La relació que caracteritza la sortida d'aquesta família de xarxes neuonals és funció dels membres de la sortida precedent. Actualitzant-se cada cop seguint la mateixa norma.

Degut a la necessitat de millorar el procés d'entrenament d'aquestes, n'apareix la variant Gated RNN's.És caracteritzen Permeten mantenir la informació en un rang temporal que va més enllà de l'estat anterior. Per tal d'aconseguir-ho, la xarxa neuronal és capaç de considerar en quin moment actualitzar el seu propi estat a partir de certes portes (unitats lògiques) que controlen el flux d'informació .

Dues de les unitats més habituals en el si de les Gated RNN són:

- Long Short Term Memory (LSTM):Permeten canviar dinàmicament l'escala de temps en que la unitat actualitza el seu estat. Les sortides de les portes depenen del procés d'entrenament de la pròpia unitat. I regulen la contribució de : la constant de temps(forget gate), el hidden layer (external input gate) i l'estat actual (output gate)

- Gated Recurrent Units (GRU) Disposen de la mateixa funcionalitat que les LSTM però la regulació de la constant de temps i l'actualització de l'estat s'encapsulen una única unitat condicional (update gate). L'update gate es responsable de determinar quina quantitat d'informació nova s'incorpora. Podent ignorar totalment el vector d'estat o per una banda, arribar a substituir-lo per un nou estat objectiu com a cas extrem. La reset gate controla quina informació de l'estat actual servirà per a calcular el nou estat objectiu.

4.3. Word Embeding (WE):

Són les tècniques en NLP que permeten que les paraules i frases del vocabulari siguin traduïdes d'un espai amb varies dimensions per paraula fins a un espai vectorial continu de menor dimensió.

4.4. Traducció de termes HPO

A fi de proveir d'un instrument capaç de dur a terme la traducció automàtica entre els termes comuns que la gent utilitza per a descriure els seus símptomes i el vocabulari fenotípic que descriu l'HPO, es va desenvolupar un instrument basat amb algoritmes de machine learning (ML).

L'estructura general de les arquitectures desenvolupades per a la traducció de termes comuns a HPO consisteixen en:

1- Expressar de la frase que expressa el layman term a un vector.

2- Obtenció d'un vector que representa un terme HPO en un espai vectorial reduït .Per mitjà d'un nivell de WE i un de RNN amb d'altres possibles convinacions.

3- Comparació d'aquest vector amb el WE dissenyat pels termes HPO. Escollint el terme HPO la representació del qual sigui més propera al resultat de la representació HPO generada.

Per mitjà de diverses combinacions de WE's i arquitectures de xarxa neuronals, s'assoleix un model capaç d'identificar correctament el fenotip en la meitat dels casos.

5. Execució de models preentrenats amb TensorFlow.js

L'objectiu central d'executar models preentrenats des del navegador rau en tres aspectes:

1. Atesa la complexitat de la tasca que s'ha de dur a terme així com la del model que la fa possible és imprescindible disposar del model entrenat i aleshores, executar la predicció.

2. TensorFlow.js permet carregar models que s'hagin entrenat amb la versió de Tensorflow o Keras en Python. Entorn en el que es va desenvolupar i es manté el model de traducció.

3. TensorFlow.js fa possible l'ús de models sense el desenvolupament d'un servidor. De manera que es poden efectuar prediccions en documents estàtics d'HTML.

El desenvolupament dels models que es presenten a continuació no és l'obtenció d'un model capaç d'efectuar de manera correcta la tasca de traducció de termes comuns a HPO.

Aquest model ja està desenvolupat. El que es pretén es validar la capacitat de les eines escollides per a poder carregar-lo i emetre prediccions en base a la petició de l'usuari.

Les característiques del model de traducció motiven les consideracions per a fer tot el procés necessari. Per una banda: preparació d'un training set, disseny d'una arquitectura i entrenament del model. Per una altra, desenvolupar l'estructura i la funcionalitat d'un aplicatiu web per tal d'utilitzar el model.

En cap cas es pretén aconseguir un model útil per a la traducció sinó que se'n volen reproduir les característiques per a poder implementar un aplicatiu capaç d'utilitzar la seva capacitat.

Tot el codi necessari per al desenvolupament del procés que es desenvolupa a continuació així com els resultats de l'entrenament dels dos models, s'adjunten a l'ANNEX A

5.1. Extracció del corpus

S'opta per generar un corpus a partir dels termes següents :

- HP:0040064 – Abnormality of limbs
- HP:0000924 – Abnormality of the skeletal system
- HP:0000707 – Abnormality of the nervous System

Tots tres pertanyents a la subontologia de Phenotypic abnormality. L'objectiu d'escollir-ne tres és per poder fer l'aproximació a la tasca de multi-classificació. D'aquesta manera es garanteix que el sistema tracti amb més d'un terme com passa amb el model existent.

La metodologia per a generar el corpus consisteix en dues etapes:

1. Generar un corpus amb tants documents com descendents tingui el terme HPO. Dels quals s'extreu name, definition, synonyms.

2. Netejar el text eliminant-ne puntuació i substituint els nombres pels seus equivalents en lletres.

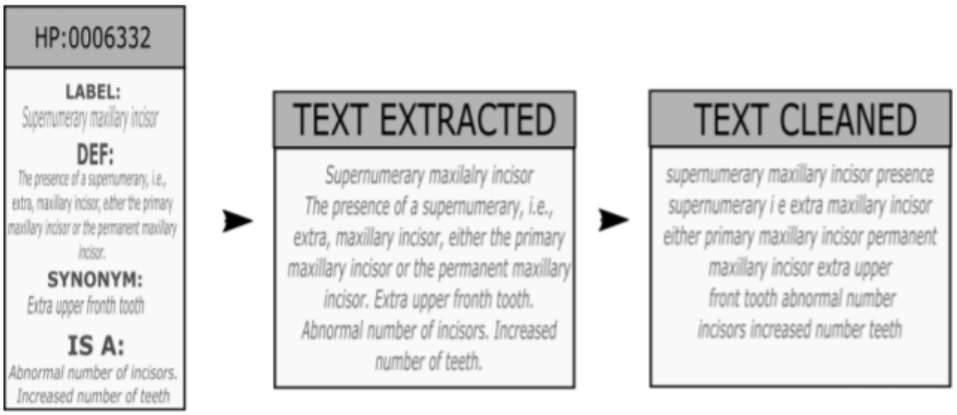

Figura 2: Exemple d'extracció i processat del document[3]

Terme	Nombre de documents
HP:0000924	3641
HP:0040064	2751
HP:0000707	2109

Figura 3: Nombre de documents per terme

Resultant el nombre de documents del corpus en 8501

5.2. Generar vocabulari i Tokenització

Les xarxes neuronals no tracten amb paraules com a entrada sinó que se les ha d'alimentar amb nombres. És per això, que cal desenvolupar un diccionari del vocabulari present en el corpus per tal de poder entrenar la xarxa neuronal.

S'opta per incloure totes les paraules presents en el diccionari però la funció emprada disposa de l'opció d'escollir una freqüència mínima. D'aquesta manera es pot escollir en quina mesura una paraula ha de ser present en el corpus per a ser inclosa en el vocabulari

Per tal d'aconseguir-ho es genera un diccionari de freqüència d'aparició dels termes del corpus. D'aquesta manera s'aconsegueix disminuir la quantitat de paraules presents en el vocabulari si es considera pertinent.

S'opta per incloure en el vocabulari totes les paraules present en el corpus . A més, s'hi inclou un terme pel padding ("pad") que correspon al 0 ja que un cop s'efectuï una predicció serà necessari que la seqüencia sigui de la mida amb la que s'ha entrenat la xarxa. Així com un per aquelles paraules que no formin part del vocabulari conegut ("desc") nombre 1.

També s'ha considerat pertinent generar dues funcions auxiliars. Una per a desar el vocabulari i l'altra per registrar quina és la frase amb més paraules. Es prendrà la llargada més gran de les seqüencies del corpus com a mida de les seqüencies que es subministraran a la xarxa neuronal per al seu entrenament

5.3. Preparació del training set

A fi de poder entrenar el model escollit cal preparar el traing set així com el seu vocabulari per quan es requereixi efectuar prediccions amb TensorFlow.js

En aquest apartat s'empren les funcionalitats anteriorment esmentades. A més, s'aplica el padding a les seqüencies del corpus amb el criteri anteriorment esmentat.

També s'opta per convertir el vector de labels de l'entrenament a una matriu per mitjà de one hot encoding abans d'alimentar el model.

```
Corpus: 8501
Frase més llarga (num): 119
Vocabulari desat a: ./vocabulari.csv
Nombre paraules en el diccionari: 6755
Mida y_train: (8501,)
Mida X_train: (8501, 119) -numtokens i max_tokens
CPU times: user 12.1 s, sys: 217 ms, total: 12.3 s
Wall time: 12.3 s
```

Figura 4: Obtenció del training set

5.4. Disseny dels models

El vocabulari del corpus obtingut és de 6755 paraules. És per això que serà necessari incloure un WE per tal de disminuir la dimensionalitat de l'entrada de la xarxa neuronal.

Com ja s'ha esmentat gran part de la tasca duta a terme en el model existent ha sigut l'elecció i desenvolupament del WE adequat per a la tasca de traducció. S'ha considerat oportú incloure'l dins del model enlloc de separar-ho i posar-lo fora.

Les dues opcions que es presenten a continuació són reduccions molt simplistes del model ja entrenat. De totes maneres, presenten les unitats existents en ell. Amb això, es pretén valorar la possible utilització d'aquest mètode com a solució.

5.4.1. Model amb LSTM:

```
Model: "sequential"
_________________________________________________________________
Layer (type)                 Output Shape              Param #
=================================================================
layer_embedding (Embedding)  (None, 119, 8)            68008
_________________________________________________________________
lstm_1 (LSTM)                (None, 119, 16)           1600
_________________________________________________________________
lstm_2 (LSTM)                (None, 119, 8)            800
_________________________________________________________________
lstm_3 (LSTM)                (None, 4)                 208
_________________________________________________________________
dense_1 (Dense)              (None, 3)                 15
=================================================================
Total params: 70,631
Trainable params: 70,631
Non-trainable params: 0
```

Figura 5: Arquitectura amb LSTM

5.4.2. Model amb GRU:

```
Model: "sequential_1"
_________________________________________________________________
Layer (type)                 Output Shape              Param #
=================================================================
layer_embedding (Embedding)  (None, 119, 8)            68008
_________________________________________________________________
gru_1 (GRU)                  (None, 119, 16)           1248
_________________________________________________________________
gru_2 (GRU)                  (None, 119, 8)            624
_________________________________________________________________
gru_3 (GRU)                  (None, 4)                 168
_________________________________________________________________
dense_1 (Dense)              (None, 3)                 15
=================================================================
Total params: 70,063
Trainable params: 70,063
Non-trainable params: 0
```

Figura 6: Arquitectura amb GRU

5.5. Entrenament dels models i conversió

Per a la realització de l'entrenament s'escullen els paràmetres següents:

- Validation_split de 0.2
- Nombre d'epochs de 15
- Batch size de 32

A més s'escull la funció loss de *categorical_crossentropy i l'optimitzador emprat és l'Adam.*

A aquestes alçades, tan sols falta guardar el model i convertir-lo al format necessari per tal de poder ser executat amb TesnorFlow.js. Per a fer-ho, s'utilitza la llibreria de TensorFlowjs de Python i es fa servir el seu convertidor per keras. De manera que s'obtenen els fitxers de format JSON i bin que permeten reproduir les arquitectures anteriorment esmentades així com la utilització dels paràmetres ja entrenats.

5.6. Aplicatiu web per a TensorFlow.js

Per tal de poder desenvolupar prediccions per a l'usuari s'ha compartimentat l'aplicatiu web en tres seccions:

- Els fitxers necessaris per efectuar-la són el vocabulari i aquells associats al model.
- L' el fitxer index.html
- El fitxer carrega.js on s'inclouen les principals funcionalitats necessàries.

5.6.1. Interacció amb l'usuari

L'objectiu de l'aplicatiu és efectuar prediccions sobre la simptomatologia que manifesti l'usuari. És per això que se l'adverteix un cop s'ha carregat el model i el diccionari de manera que el navegador ja està llest per efectuar prediccions.

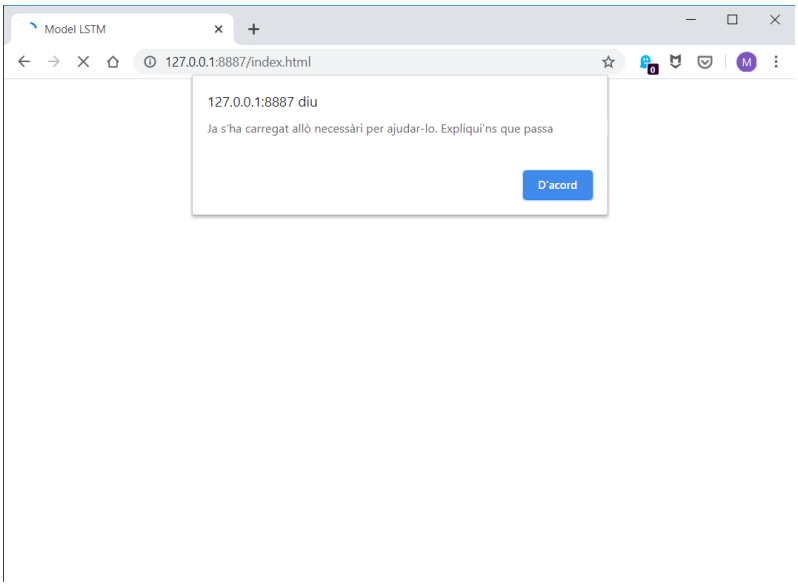

Figura 7: Avís de model carregat

A continuació, es mostra la pantalla amb el quadre de diàleg per a introduir el text. De manera que un cop s'envia el model efectua la predicció i mostra la probabilitat associada en la predicció i el terme corresponent a aquesta.

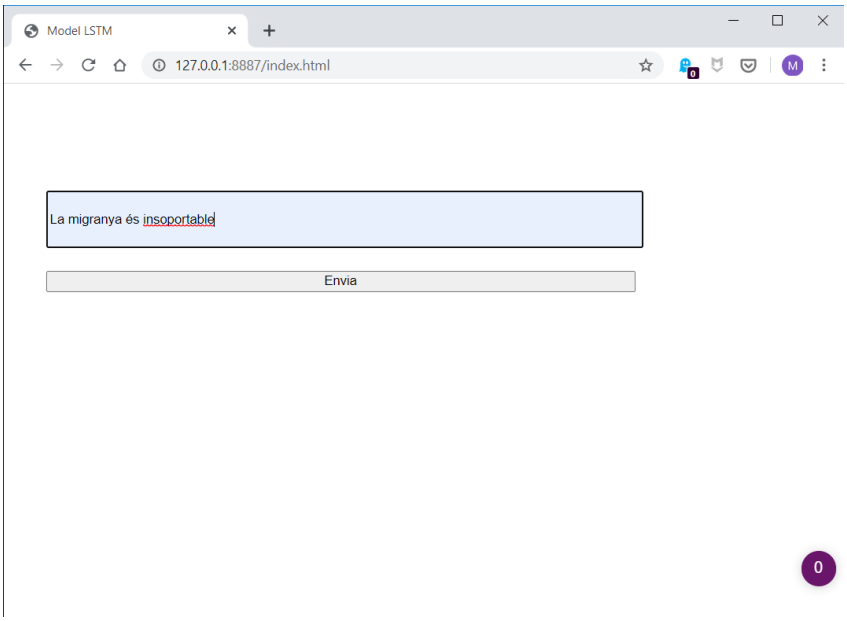

Figura 8: Exemple diàleg usuari

De manera que l'usuari visualitza finalment els termes HPO i les probabilitats que el model els hi atorga així com el llindar escollit.

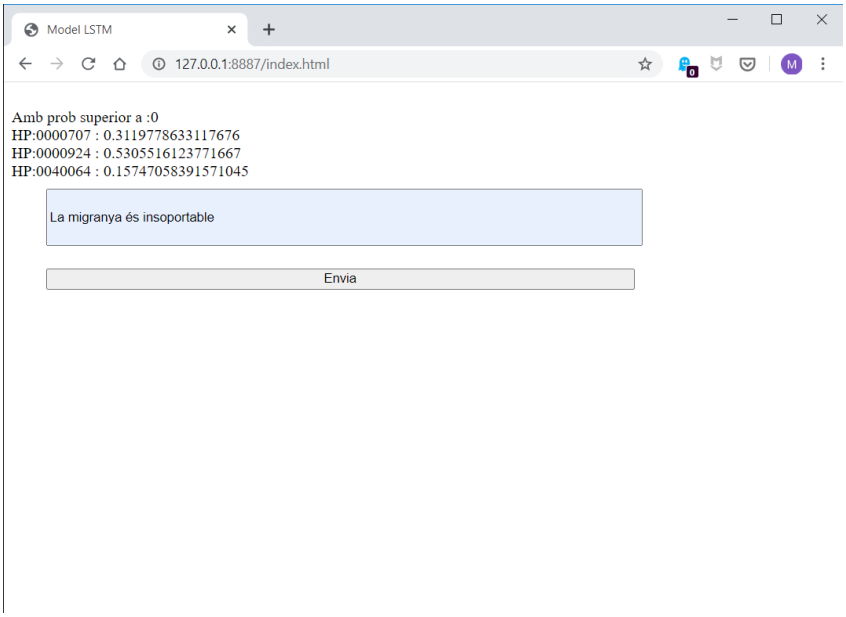

Figura 9: Mostra de prediccions del model

5.6.2. Funcionalitat desenvolupades

5.6.2.1. Càrrega del model i el vocabulari:

Per tal de poder efectuar les prediccions es necessari que es carreguin tant el model com el diccionari. El temps requerit per a efectuar aquest procés és rellevant pel que fa a la interacció amb l'usuari i s'aprecia la validesa de les eines escollides ja que no suposa cap inconvenient

5.6.2.2. Preprocessament i tokenització:

A diferència de l'especial consideració que s'havia tingut en aquesta etapa per la preparació del training set, en tractar amb l'usuari, s'espera que utilitzi termes corrents. És per això que l'exigència en aquesta etapa disminueix notablement. S'opta per eliminar puntuació i convertir a lletres minúscules.

A continuació, es preparen les seqüencies per tal de poder-se subministrar a la xarxa neuronal. En aquest moment, es genera la seqüencia en base a les paraules i a continuació

s'efectua el padding.

5.6.2.3. Predicció:

Al disposar de la descripció introduïda per l'usuari en les condicions requerides per l'usuari ja pot subministrar-se al model i efectuar la predicció. En aquest cas també és important el temps necessari per efectuar-la. De la mateixa manera que en la càrrega del model, en aquest cas, resulta una bona aproximació.

5.6.2.4. Tracte de la resposta:

La resposta del model, és una llista de probabilitats. Es va dissenyar d'aquesta manera per tal de poder desenvolupar la funcionalitat que es requeriria si es tractes d'una llista amb molts més elements com passa amb el model existent.

Per traslladar la informació a l'usuari es creen dues alternatives. Per una banda la possibilitat de subministrar la més provable i per una altra escollir la quantitat a partir d'un llindar predeterminat.

5.7. Limitacions i canvi d'enfoc

Malgrat haver-se arribat a una primera aproximació en termes d'aplicatiu web per suplir les necessitats que motiven aquest projecte, no s'aconsegueix carregar el model amb layers GRU.

La limitació en d'aquest enfoc radica en la complexitat de l'arquitectura de la xarxa neuronal i en l'elecció d'incloure l'WE en l'interior del model. Com s'ha presentat anteriorment les estructures GRU presenten la porta Update i la reset. Cal especificar-ne l'attention vector en la seva definició si després vol poder efectuar-se prediccions amb Tensorflow.JS.

L'enfoc que s'havia desenvolupat fins ara resultava molt atractiu perquè es disposava de la potencialitat d'incloure el programari necessari (llibreries com Tensorflow.js) sense la necessitat d'instal·lar-lo. A més, podia realitzar-se des la banda del client de manera que no hi havia necessitat d'accedir a un servidor extern per a efectuar les prediccions.

De totes maneres, el model existent és de molt major complexitat que els dos que s'han presentat en aquesta secció. A més, es subjecte a variacions en l'arquitectura i modificacions de les diferents parts que el conformen. De manera que es considera de major utilitat optar per un canvi de paradigma basat en dos eixos.

Primerament, el model que efectuarà la petició agafarà el text sense cap mena de processat previ . És a dir, serà processat i tractat de la manera necessària per efectuar la predicció (en la banda servidor). Però aquesta, es desvincula de l'enfoc d'aquest projecte més enllà d'emetre la informació segons els protocols establerts.

En segon lloc, s'haurà de crear una comunicació per tal de poder unir la banda del client amb la del servidor que està pendent per poder efectuar les prediccions.

6. Mètode de visualització dels termes HPO:

Un dels eixos centrals d'aquest projecte és poder millorar l'intercanvi d'informació clínica especialitzada i qualsevol persona.

Es pretén fer un estudi de les possibilitats existents per a la visualització d'ontologies i concretar-ne la seva adaptabilitat als nostres requisits La concreció de la tasca que es pretén desenvolupar així com la immensa varietat d'eines i entorns de visualització existents requereixen d'un desenvolupament minuciós del tema.

6.1. Producció amb esquema existent de vega i aplicatiu

En un primer terme i atès que el tracte amb la informació de l'ontologia és duia a terme amb Python. Va semblar raonable utilitzar un esquema ja existent amb Vega. Va ser més una eina que va sorgir en un moment que es requeria una major familiarització amb l'ontologia i la programació web.

Vega és una gramàtica de visualització i llenguatge declaratiu que permet crear, guardar i compartir dissenys de visualització interactiva.

Aquest és un dels aspectes que en un primer terme va ser considerat rellevant ja que resulta imprescindible garantir la interacció per a transmetre i captar informació.

A més, a partir del format JSON ja es poden generar vistes basades en la web. Punt especialment rellevant.

6.1.1. Preparació de dades i elecció de l'esquema

Per tal de poder gaudir d'esquemes molt desenvolupats i abans d'entrar a estudiar-ne l'estructura cal cenyir-se a l'estructura de les dades que requereixen.

A fi de poder sistematitzar l'ús de l'eina es creen funcions per a la creació de l'esquema pertinent així com la preparació de fitxers en formats JSON que posteriorment seran visualitzats amb l'aplicatiu web. Les funcionalitats necessàries escrites en Python s'adjunten a l'Annex B

S'opta per utilitzar el radial tree layout perquè:

- Representa la profunditat dels nodes fills amb colors.
- Mostra el label (id HPO)

- Permet veure el nom en col·locar el ratolí sobre els nodes

6.1.2. Aplicatiu

La funcionalitat de l'aplicatiu consisteix en permetre a l'usuari escollir quina de les cinc subontologies que configuren l'HPO vol visualitzar.

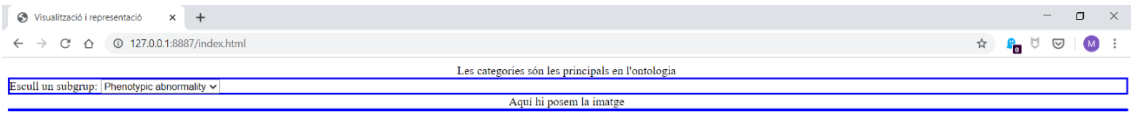

Figura 10: : Aplicatiu per a la visualització amb vega

Es tracta doncs de la primera aproximació a la relació entre un terme HPO proposat a l'usuari segons el seu nom i una visualització del seu context en l'ontologia.

6.1.3. Resultats

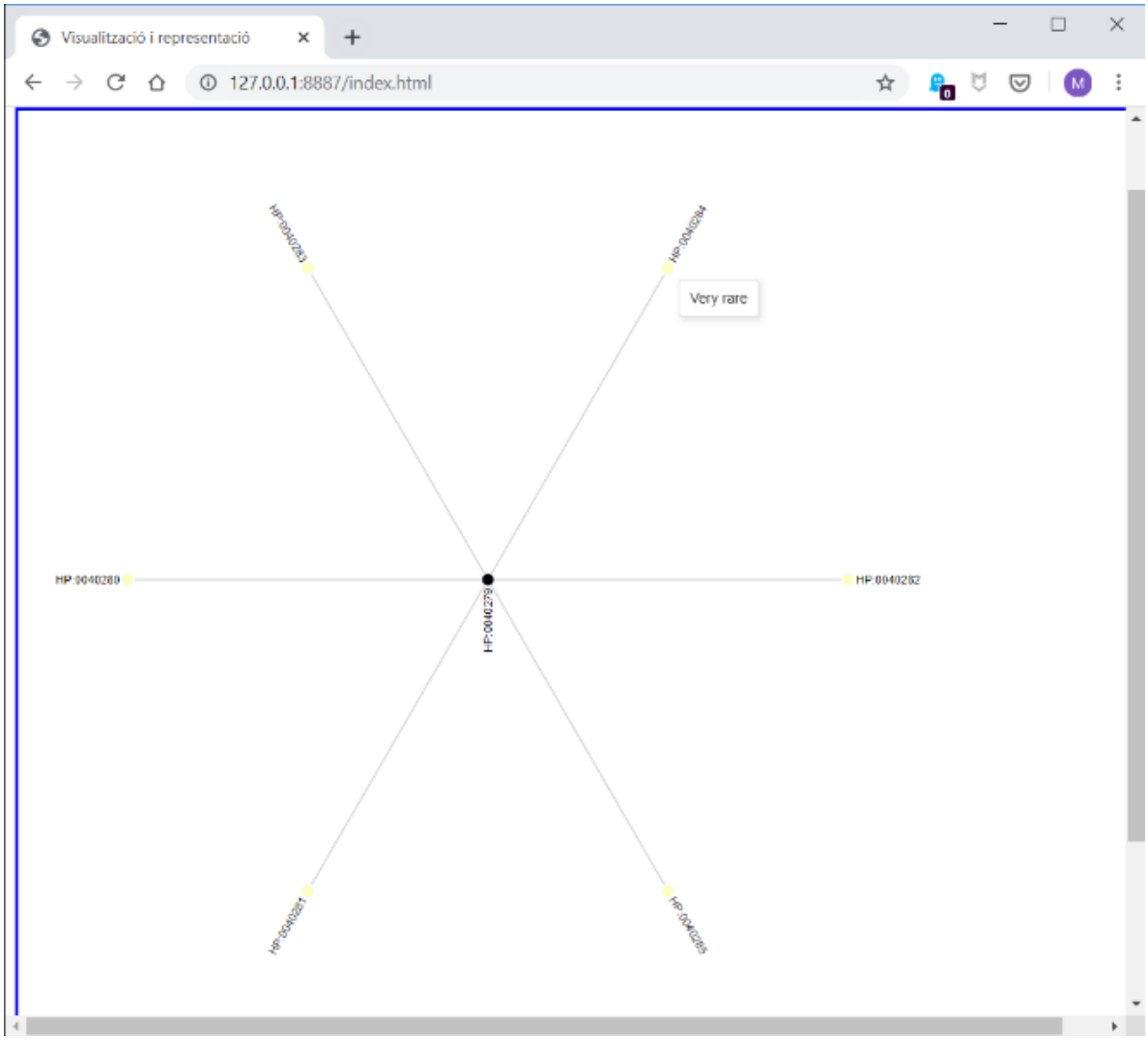

Figura 11: Visualització de frequency

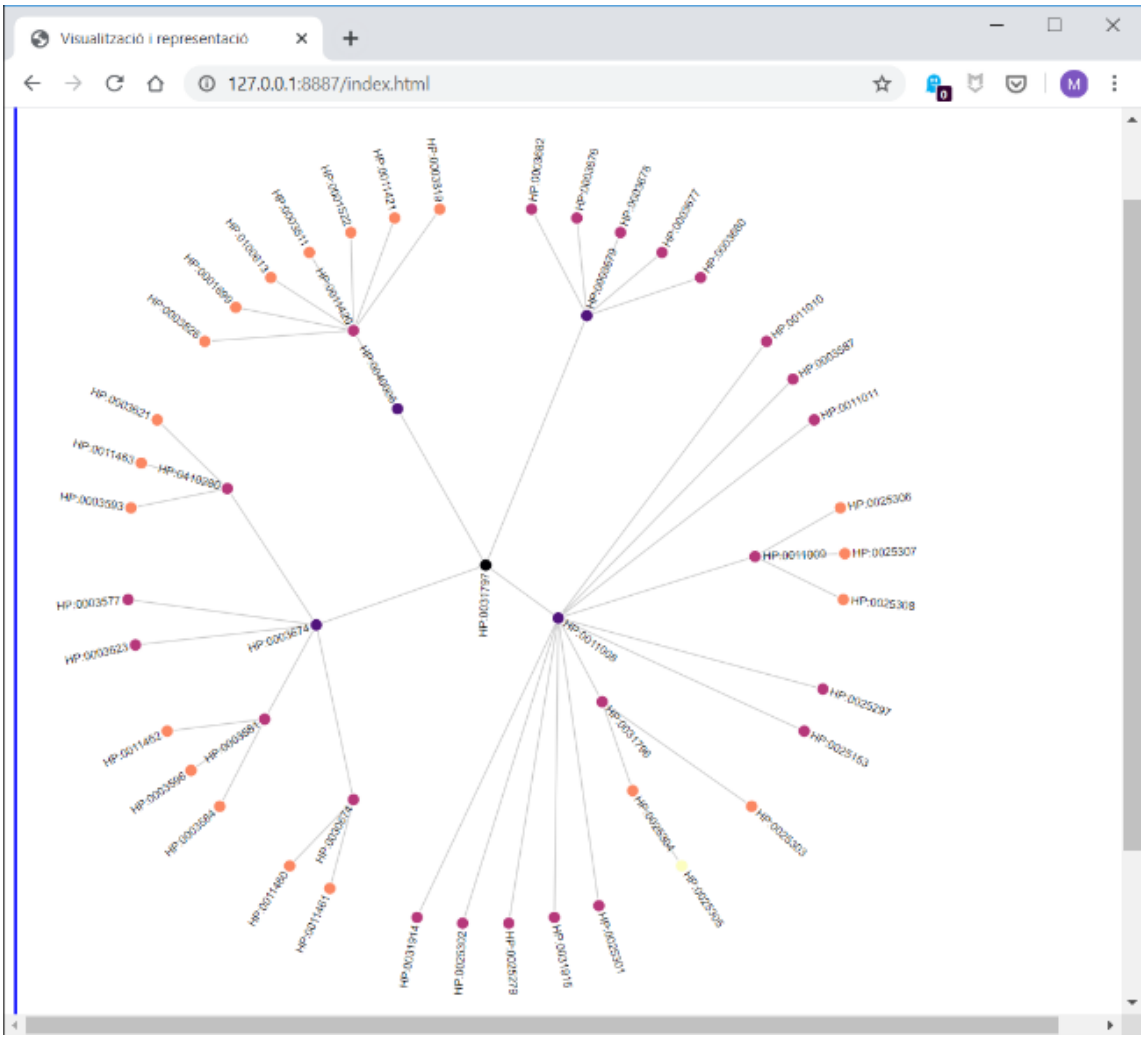

Figura 12: Visualització de Clinical Course

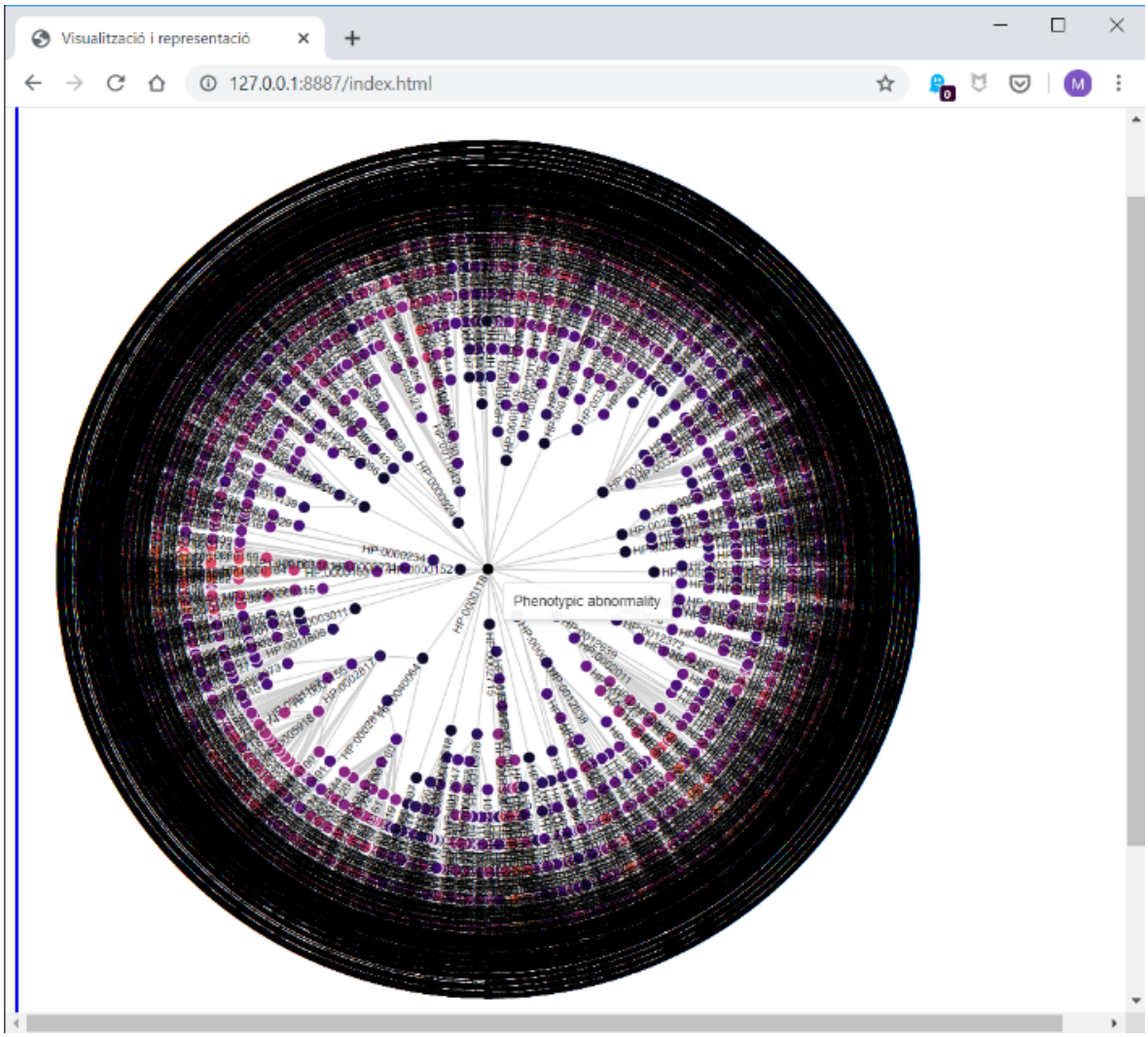

Figura 13: Visualització de Clinical modifier

S'aprecia que la informació resulta accessible si la quantitat de nodes no és elevada (fins a 300). En cas de ser superior ja comença a ser difícil apreciar la informació que transmet el graf. En ampliar-se més encara la quantitat de nodes es perd totalment la capacitat d'accedir a la informació disponible.

Per tal d'incrementar la qualitat de la informació que s'ofereix a l'usuari i facilitar que la concreti segons les seves necessitats es decideix:

- Modificar l'estructura radial:
- Visualitzar el label enlloc de l'id d'HPO
- Emprar una llibreria especialitzada en la visualització d'ontologies

6.2. Visualització amb Ontospy

Ontospy és una llibreria de Python desenvolupada per la inspecció i la navegació de vocabulari codificat utilitzant la semàntica W3C. El seu ús consisteix en carregar un graf instanciant-lo a un fitxer que conté una definició RDFS o OWL. De manera que es pot retornar un objecte que permet accedir a l'ontologia.

La llibreria disposa de nombroses funcionalitats per a la visualització d' ontologies. A partir de l'estudi del seu funcionament, s'arriba a la conclusió que per a poder integrar la voluntat d'oferir una resposta gràfica a l'usuari serà més convenient optar per la llibreria en que OntosPy es basa d3.js

6.3. Visualització amb D3.js i aplicatiu

D3 deu el seu nom a "Data-Driven Documents" sintetitza de manera notable la voluntat dels seus autors per a oferir una llibreria en JavaScript que permet crear visualitzacions dinàmiques i interactives prenent com a base dades que a través de qualsevol navegador web poden ser visualitzades.

Un altre aspecte que resulta clau per a l'elecció de D3.js és el fet de tenir un mòdul dedicat exclusivament a la representació de jerarquies (d3-hierarchy).

6.4. Elecció de la representació gràfica:

Com s'ha esmentat anteriorment el graf radial no resultava satisfactori per a l'objectiu de facilitar el context a l'usuari.

Els objectius de la representació gràfica és d'augmentar la interacció que fins ara s'havia assolit amb l'usuari i són:

- Permetre que l'usuari esculli els nodes que li interessin
- Augmentar la quantitat de nodes visibles a partir de l'elecció de l'usuari
- En un primer terme visualitzar exclusivament els nodes fills del terme HPO subministrat

Dels marcs existents les dues opcions preferibles són el Tidy Tree i el clúster dendogram. Es considera més adient el segon perquè malgrat suposar una representació menys compacta, presenta els nivells de profunditat dels nodes a una mateixa alçada. De manera que

s'aconsegueix incorporar de manera accessible la informació present en l'estructura de l'ontologia gràficament.

A més, es vol incloure l'opció d'anar ampliant o disminuint la quantitat de nodes presents en base a la interacció amb l'usuari. En altres termes, es vol que el graf sigui desplegable i s'inicialitzi amb únicament els nodes fills i l'arrel.

S'escull doncs una implementació [4] en que la profunditat dels nodes es calcula amb la distància a l'arrel. Per tal d'oferir una disposició eficient i ordenada dels nodes s'usa la funcionalitat de la llibreria que ho dur a terme a partir de l'algoritme Reingold-Tilford.

6.4.1. Preparació de dades

Per tal d'ajustar la informació de l'ontologia per poder ser carregada i visualitzada s'ha d'estructurar la informació de manera que per cada node descendent del terme HPO previst s'especifiquin:

- Id: el número que identifica el terme HPO
- Name: El nom que se li dona al terme
- Children: El descendent del node en qüestió
- parentID: el número identificariu del terme HPO del que és descendent el node.

Tan sols pot haver-hi un node arrel i serà interpretat com a tal si no s'especifica quin és el node pare i l'estructura serà una llista d'objectes.

Resulta necessari modificar la funció existent per a la creació de l'estructura jeràrquica amb el mètode *stratify* com es mostra a continuació:

```
const root = d3.stratify()(data);
```

6.4.2. Aplicatiu

La funcionalitat de l'aplicatiu es tan sols de visualització ja que serà a través de la resposta de predicció del model que s'obtindrà la visualització que s'està desenvolupant.

S'ha incorporat la funcionalitat de desplegar els nodes fills d'aquell node que es vulgui així com replegar-los novament.

6.4.3. Resultats

S'assoleix la necessitat de visualitzar els nodes fills del terme HPO escollit i que sigui la interacció de l'usuari que desplegui més segons consideri oportú.

Figura 14: Visualitzacio del dendograma

S'ha aconseguit incloure la capacitat d'augmentar la informació disponible segons el criteri de l'usuari. Es disposa dels noms enlloc dels termes HPO com s'havia considerat pertinent.

Cal ampliar les funcionalitats presents en la representació perquè en no poder-se desplaça el graf s'arriba a un punt en que els nous nodes desplegats no són identificables. S'aconsegueix però que els respectius nivells de profunditat es visualitzin a un mateix nivell.

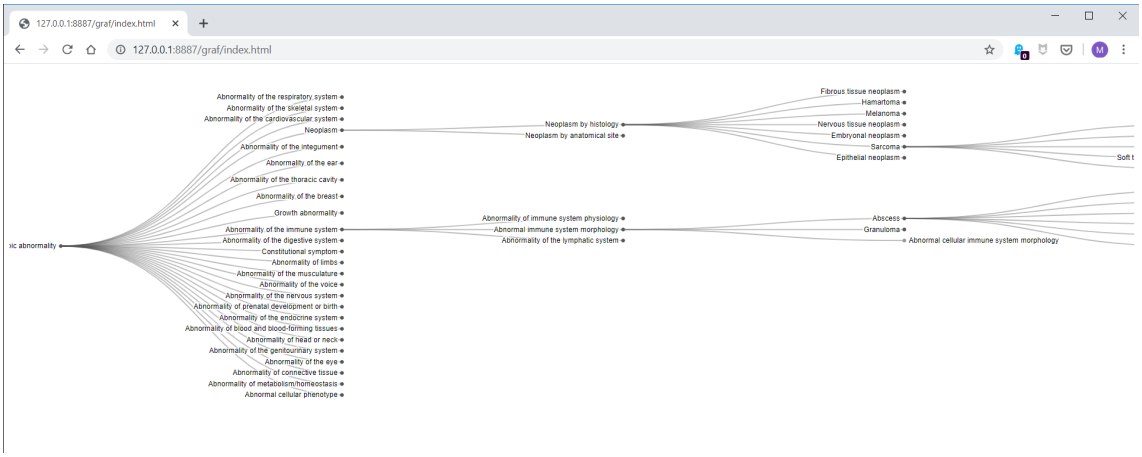

Figura 15: Visualització dendograma desplegat

6.5. Modificació d'estructura

S' escull una versió de dendrograma[5] que incorpora les funcionalitats següents:

- Desplegable: Permet ampliar la informació present disponible així com reduir-la prement als nodes.
- Arrossegar i deixar anar: Navegant en la visualització es pot accedir a tots els elements presents.
- Ampliable: Es pot centrar l'atenció en una secció concreta o contràriament ampliar el marc visible reduint-se la mida dels elements.
- Dimensionament automàtic: En desplegar-se més informació s'ajusta de manera automàtica la informació disponible al marc visible.

6.5.1. Preparació de dades

L'estructura de les dades subministrades a la que en genera la visualització consta dels següents punts:

- Id: Identificador del terme HPO

- Name: Nom del terme HPO
- Children: Llista de nodes descendents amb l'id, nom i descendents.

A fi d'aconseguir generar-ho s'opta per en un primer terme obtenir la informació relacional de la informació en els termes esmentats en l'anterior secció i seguidament obtenir de manera recursiva l'estructura desitjada per a generar el fitxer JSON.

6.5.2. Funcionalitat de registre de clics

Per tal de poder incorporar la informació que l'usuari considera pertinent es crea un registre de la interacció amb la visualització. D'aquesta manera, s'aconsegueix una llista amb tots els nodes que han interessat a l'usuari.

L'objectiu és constatar la capacitat de dur a terme aquesta tasca per poder en un altre estadi entrenar el model amb aquesta informació.

6.5.3. Resultats

Un cop es desplega certa quantitat de nodes la informació present s'ajusta segons les necessitats.

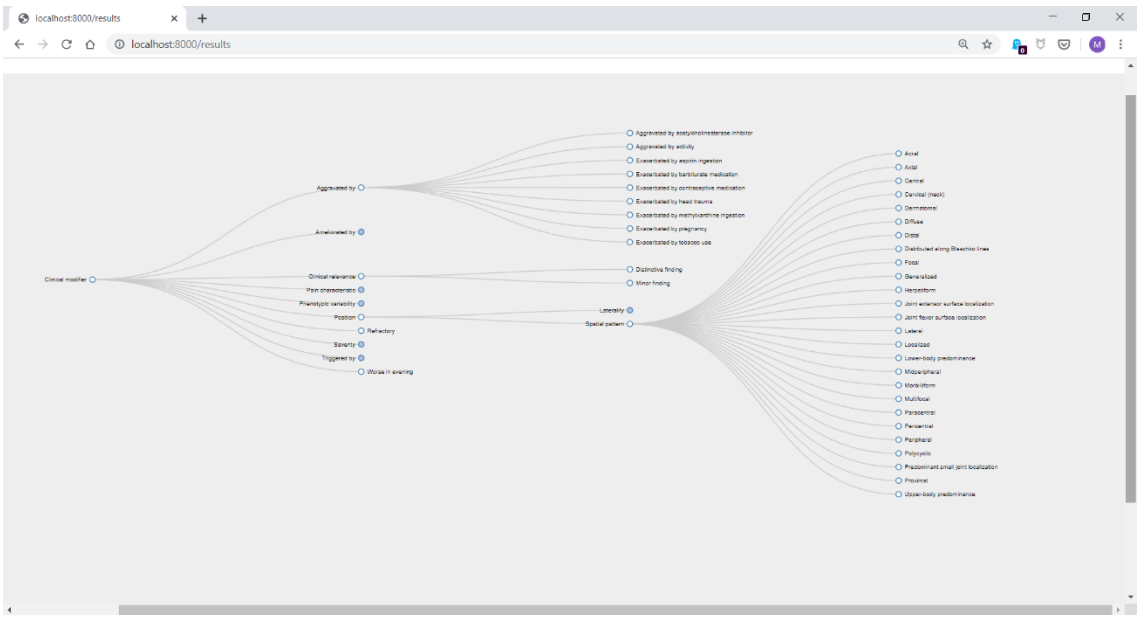

Figura 16: Aplicatiu amb dendograma desplegat

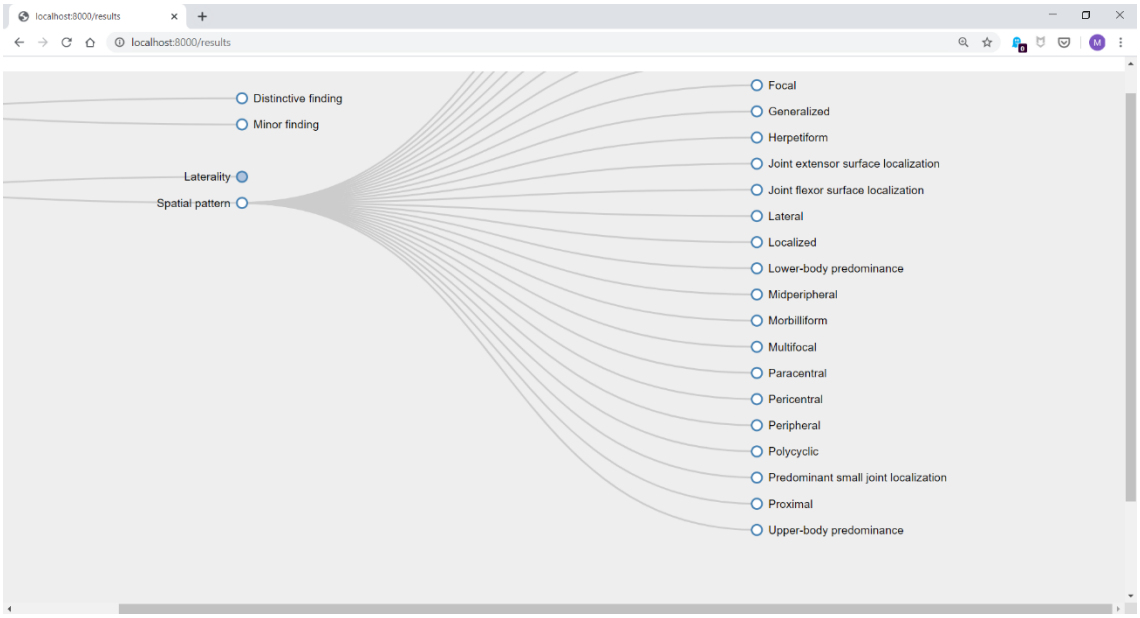

Figura 17: Exemple d'ampliació del rang visible

7. Proposta d'aplicatiu:

En haver-se pres la decisió d'externalitzar la funcionalitat de traducció de termes comuns a HPO, resulta necessari generar un aplicatiu capaç de comunicar-se amb el servidor que dugui a terme les prediccions. A més, cal integrar els dos eixos vertebradors de l'aplicació:

- Recaptar els layman terms de l'usuari i enviar-lo al model perquè emeti la predicció
- Rebre les prediccions i oferir la visualització

A tal fi , es decideix desenvolupar un aplicatiu amb Node.js i Express.

7.1. Entorns escollits

7.1.1. Node.js

Entorn de programació de la banda del servidor web orientada per esdeveniments. Basat en el motor de JavaScript V8. JavaScript és un llenguatge de programació basat en objectes tradicionalment utilitzat en la banda client. Un tret que caracteritza Node és haver entaulat la comunicació entre servidor i client per mitjà del mateix llenguatge. A més, també es distingeix per delegar totes les crides a un bucle d'esdeveniments que pot executar múltiples sub-processos (*non-blocking*)[6]

7.1.2. Express:

És un entorn de treball per aplicacions web per a Node. Subministra un conjunt de característiques per a l'implementació de pàgines web.

7.2. Estructura de l'aplicatiu

A continuació es mostren els directoris i fitxers que vertebren l'aplicació.

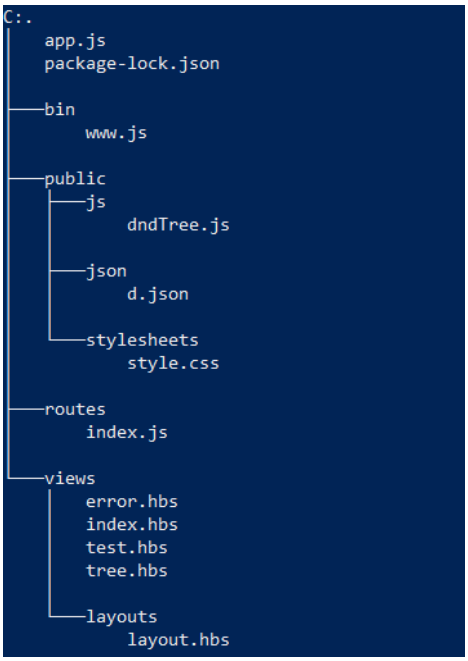

Figura 18: Estructura dels fitxers

7.2.1. Directori bin

És la ubicació on es poden definir diversos fitxers d'inicialització. En concret el fitxer www.js permet iniciar l'app expres com a servidor web.

Aquest estil és característic d'express 4.0 i serveix per mantenir i actualitzar de manera independent el servidor. Amb versions anteriors s'especificaven en un mateix fitxer les configuracions del port així com les rutes.

En el fitxer www.js doncs s'hi especifiquen les dependències dels mòduls, a continuació s'hi crea i especifica el port. A més, es crea el servidor HTTP i és genera un gestor d'esdeveniments per tal d'identificar possibles errors en la inicialització del servidor.

7.2.2. Fitxer app.js

És el cos de l'aplicació en ell s'hi defineixen tots els elements que resultaran necessaris en termes de funcionalitat per poder finalment exportar l'objecte app .

S'hi especifiquen:

- View engine: Per tal de sistematitzar la producció s'empra handlebars. És un llenguatge de plantilla sense lògica que mantenen les visualitzacions i el codi separat.

- Directori públic : Al subministrar el nom del directori es permet gestionar tot el contingut estàtic (fitxers CSS, JavaScript...) de manera directa. En concret seran l'estil (fitxer d.json) i la funcionalitat (dndTree.js) que fan possible visualitzar el graf.
- Rutes: Serà definit en el fitxer index.js i consisteix en determinar com l'aplicació respon a les peticions del client a través d'una ruta URI i un mètode de petició sigui GET, POST...

7.3. Aplicatiu

S'opta per a compartimentar les dues funcionalitats a través de peticions HTTP independents.

Per tal d'establir la comunicació entre client i servidor s'acorden i estandarditzen les estructures JSON per a garantir que l'intercanvi d'informació tant facilitada com rebuda es duguin a terme només si es compleixen els requisits determinats facilitant d'aquesta manera l'automatització de la lògica del sistema presents en l'annex D.

- Translàtor-api-input_schema.json: S'hi inclou el text introduït per l'usuari i la quantitat màxima de prediccions associades que es volen rebre

- Translàtor-api-output_schema.json: S'hi inclou l'estructura de dades per a generar la representació gràfica de les prediccions. És genera en el servidor que efectua la predicció a partir de les dades facilitades per l'usuari.

7.3.1. Pàgina d'inici

La pàgina d'inici consta d'un requadre per tal que l'usuari pugui introduir el text i enviar-lo al servidor extern que resta a l'espera per a emetre prediccions.

7.3.2. Comunicació client servidor

En un primer terme s'opta per a realitzar una petició POST per a permetre la comunicació client-servidor. Per tal de garantir que les dades s'ajustin al format establert, aquestes es visualitzen a la pàgina.

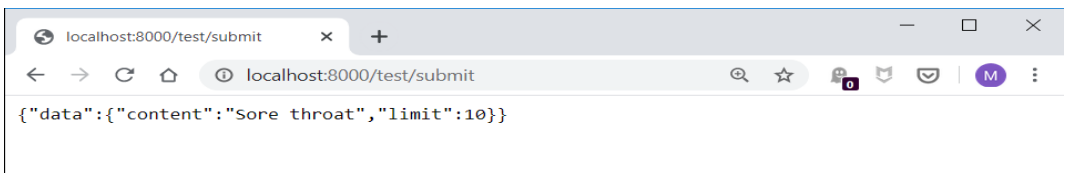

Figura 19: Exemple de JSON enviat

Finalment, es considera oportú modificar la sol·licitud POST per una de GET. Això es motivat pel fet que la petició POST no espera una resposta. De manera que en fer un GET s'aconsegueix gestionar la petició de manera asíncrona. No obstant, atès que no s'empra el servidor extern un cop l'usuari introdueix el text es redirigeix la petició a la petició que està a l'espera de la resposta.

```
1. router.get('/test/submit',async function(req, res, next) {
2.   res.redirect('/results')
3. });
```

7.3.3. Comunicació Servidor Client

A fi de dur a terme la comunicació servidor client s'opta per una petició GET. Aquesta resta a l'espera de rebre la resposta del servidor per a redirigir l'usuari al graf dels termes HPO.

Per tal de reproduir l'escolta i la petició del client es penja un fitxer JSON amb l'estructura desitjada en un repositori de Github amb l'objectiu de reproduir el comportament que hi hauria amb el servidor que ha de dur a terme les prediccions. D'aquesta manera, es pot comprovar el comportament de l'aplicació al realitzar peticions (duta a terme amb una petició) a servidors externs.

La informació rebuda es processa i es genera un fitxer estàtic per tal de poder dur a terme la visualització del graf.

```
router.get("/results", async function(req, res,next){
  request('https://raw.githubusercontent.com/maxmontoliutorruella/server/master/sample.json',function(err,response,body){
    console.log('error',err)
    console.log('statusCode:', response && response.statusCode)
    var parsedBody = JSON.parse(body)

   console.log('Get del server',parsedBody.data)


  //Escrivint al json
   fs.writeFile("./public/d.json", JSON.stringify(parsedBody.data), err => {

     // Checking for errors
     if (err) throw err;

     console.log("Done writing"); // Success
   });
  res.render('tree')
 });
});
```

L'aplicatiu que es descriu a continuació és disponible en el repositori de Github següent:

https://github.com/maxmontoliutorruella/TFG-Aplicatiu/tree/master

Els requisits necessaris es troben en el fitxer packages.json. Per a la seva execució cal estar en el directori pertinent i cridar la comanda npm start. Aleshores l'aplicatiu ja resultarà accessible en el port local 8000.

7.4. Propostes de millora

Generalitzar les metodologies actuals per a generar més gràfics d'altres propostes HPO. D'aquesta manera s'ampliaria tant la quantitat d'informació oferta a l'usuari com la potencialment recaptable través de la seva interacció.

Valorar la capacitat de recol·lectar informació per a la seva incorporació en l'entrenament del model existent. Si a més s'inclogués una validació de l'usuari podria arribar a automatitzar-se la incorporació de la informació a la ja present.

Finalitzar l'etapa de posada en marxa de l'aplicatiu i d'aquesta manera fer-lo accessible a tots aquells usuaris que poden beneficiar-se'n. Al haver emprat Node com a entorn facilita imaginar-se l'accés de la solució a l'entorn actual. Es pot executar des de qualsevol navegador de manera que

esdevé accessible a qualsevol dispositiu amb accés a internet i navegadors web. A més, hi ha eines que permeten integrar els aplicatius de Node a aplicacions mòbil

Estendre les potencialitats del model i la seva integració en l'aplicatiu a d'altres llengües. La tasca de traducció que efectua el model existent és de l'anglès a termes HPO. En cas de desenvolupar-se d'altres models capaços de reproduir la tasca de traducció amb d'altres llengües s'ampliaria la quantitat d'usuaris que podrien treure profit de l'eina desenvolupada.

8. Impacte econòmic i mediambiental

8.1. Impacte econòmic

Els costos requerits per al desenvolupament del projecte es divideixen en 3 categeories:

- Costos personals: Els costos personals són els associats a les prestacions salarials dels membres implicats: Estudiant en pràctiques fent mitja jornada durant tot el projecte. Així com la remuneració de tres supervisors. L 'assessorament de l'equip s'estima a 120€/h
- Materials: Els materials emparats són únicament un ordinador portàtil (Acer Aspire 5 A515-52, 650 €) se n'estima una vida útil de tres anys resultant en 18[€/mes]
- Costos energètics: El consum elèctric de l'ordinador portàtil s'estima en 0.3 KWh. El seu ús és de 4 hores al dia i en considerar el preu de l'electricitat 0.2 €/KW

D'on resulta un cost total del projecte de 9142 €. Tal i com es detalla a continuació.

Empleat	**Salari**	**Temps de treball [h/setmana]**	**Cost [€/mes]**
Becari en pràctiques	8 [€/h]	20	640
Equip assessor	120 [€/h]	1	480
Total			8960

Taula 1: Cost del personal

Item	Energia[KWh]	Us [h/dia]	Cost [€/mes]
Ordinador portàtil	0.3	4	4.8
Total			38.4

Taula 2: Cost del material

Elements	Cost mensual [€/mes]	Cost projecte [€]
Cost personal	1120	8960
Material	18	144
Cost energètic	4.8	38.4
Total		9142

Taula 3: Cost total

8.2. Impacte mediambiental

L'impacte mediambiental derivat d'aquest projecte és l'equivalent al consum d'energia de l'ordinador necessari per a la seva execució.

Aquest consum, tal i com s'ha explicat en el punt anterior, equival a l'ús de l'ordinador durant 4 hores / dia durant els dies laborables dels 8 mesos que dura el projecte. El consum de l'ordinador s'estima en 0,3 kWh, el que equival a **12 kWh** consumits al llarg de tot el projecte.
S'estima que cada kWh generat de fonts d'energia no renovables es tradueix en 0,07 Kg de CO_2 per kWh, resultant en un total **0,84 Kg de CO_2** emesos per a la producció total dels 8 mesos de durada del projecte.

Conclusions

Malgrat haver-se desenvolupat un aplicatiu capaç d'executar prediccions des del navegador mateix, s'arriba a la conclusió que és necessària la separació client servidor per a dur-la a terme.

Per mitjà d'un estudi sobre eines de visualització, s'assoleix l'objectiu de facilitar la comprensió d'estructures d'informació complexes com són les antologies en un entorn web.

Es permet a l'usuari considerar quina informació li resulta de més utilitat. A més, s'incorpora la capacitat d'ampliar l'eficiència dels sistemes ja existents per mitjà de la incorporació de la informació recol·lectada en el si de la interacció de l'usuari amb l'aplicatiu.

S'aconsegueix desenvolupar un aplicatiu web capaç d'utilitzar un model encarregat de la tasca de traducció entre els símptomes expressats per a gent sense coneixements mèdics i la terminologia HPO.

Es basteixen doncs, les bases de la solució desitjada per tal d'aprofitar la globalitat d'internet, l'accessibilitat a la informació que ofereixen els navegadors web i la potencialitat de sistemes de NLP i eines de visualització per a fer-la comprensible.

Agraïments

Àlex , Jon i Enrico gràcies per la paciència i guia

Pensant en “No es tracta aquí de trobar la immortalitat sinó de donar un cert valor al que és mortal “. Us vull agrair a vosaltres, tots els imprescindibles, de donar-li tant de valor al que és mortal.

Bibliografia

Referències bibliogràfiques

[1] *Köhler S, Doelken SC, Mungall CJ, Bauer S, Firth HV, Bailleul-Forestier I, et al. (January 2014).* "*The Human Phenotype Ontology project: linking molecular biology and disease through phenotype data*". *Nucleic Acids Research. 42 (Database issue): D966-74.*

[2] D. MAN, "*Didactica mathematica*", Vol. 31(2013), No 1, pp. 43–46

[3] Manzini, Enrico "*Automatic trnslation between layman and HPO terms using Machine learning algoriths*", (July 2014)

[4] M. Bostock "*Collapsible Tree*", (October 2018), [https://observablehq.com/@d3/collapsible-tree 01-06-20]

[5] R. Schmuecker "Block 79226762", (May 2019) [http://bl.ocks.org/robschmuecker/7926762 01-06-20]

[6] E. Brown "Web Development with Node & Express" (July 2014) pp 1-15

Bibliografia complementària

Les referencies esmenades en aquest apartat no s'han referenciat de manera explícita en la memòria però s'han consultat i utilitzat per fonamentar els coneixements reflectits

Per a la definición dels termes de l'estat de l'art

[7] E. Brown "Web Development with Node & Express" (July 2014) pp 1-15

Per a la comprensió del funcionament de TensorFlow.js

[8] https://www.coursera.org/learn/browser-based-models-tensorflow