jamk

Ontologiakartan rakentaminen Wikidatan avulla

Jussi Hurtta

Toukokuu 2022

Tietojenkäsittely ja tietoliikenne

Insinööri (AMK), tieto- ja viestintätekniikka

jamk | Jyväskylän ammattikorkeakoulu
University of Applied Sciences

jamk

Kuvailulehti

Hurtta Jussi

Ontologiakartan rakentaminen wikidatan avulla

Jyväskylä: Jyväskylän ammattikorkeakoulu. Toukokuu 2022, 36 sivua.

Tekniikan ala. Tieto- ja viestintätekniikan tutkinto-ohjelma. Opinnäytetyö AMK.

Julkaisun kieli: suomi

Julkaisulupa avoimessa verkossa: kyllä

Tiivistelmä

Opinnäytetyössä tutkittiin pystyykö asiasanalistalle rakentamaan hierarkiagraafin Wikidatan julkisen ontologian avulla. Tavoitteena oli muokata litteästä listasta BERT-mallinnukseen käytettyille termeille hierarkiarakenne, jolla asiasanoja voisi luokitella.

Työ rakennettiin AWS-pilveen ja dataa kerättiin Wikidatan rajapinnoista Pythonin ja SPARQL-kyselykielen avulla. Dataa tallennettiin Neptune-graafitietokantaan käyttäen Pythonia ja Gremlin-kyselykieltä. Graafitaulua tarkasteltiin Jupyter-notebookin työkalujen avulla.

Työn lopputulos koostui Neptune-graafikannasta ja siihen tallennetuista entiteeteistä ja niille löydetyistä suhteista. Todettiin, että Wikidatan hierarkiat suurilta osin eivät olleet peilattavissa käytettävään asiasanasastoon. Käytettävä asiasanasto ja sille halutut entiteettisuhteet poikkesivat liikaa Wikidatan yleisestä ontologiasta.

Käytetyt metodit olisivat saattaneet olla käyttökelpoisempia, jos asiasanasto olisi muistuttanut yleistä ontologiaa enemmän. Graafirakenne osoittautui hyväksi tavaksi tallentaa ontologiahierarkiaa, mutta varsinaiset entiteettisuhteet jouduttiin hakemaan muista lähteistä.

Avainsanat (asiasanat)

BERT, Graafitietokannat Gremlin, Koneoppiminen, Neptune, Ontologia, Python, SPARQL, Tietokannat

jamk

Description

Hurtta Jussi

Building an ontology chart with Wikidata

Jyväskylä: JAMK University of Applied Sciences, May 2022, 36 pages.

Information and Communication Technology. Bachelor's Degree Programme in Information and Communication Technology. Bachelor’s thesis.

Permission for open access publication: Yes

Language of publication: Finnish

Abstract

The topic of research was to find out if Wikidata’s ontology could be used to build a hierarchical graph for an unrelated list of subject tags. The goal was to modify an existing flat list of classes used for BERT-modeling into a hierarchical structure with which the tags could be categorized.

The project was built on AWS cloud infrastructure, while Python and SPARQL were used to collect data from Wikidata API:s. The collected data was transferred to a Neptune graph database using Python and the Gremlin query language. The graph was examined using Jupyter notebooks.

The final product consisted of a Neptune-graph database containing the entities and relationships found for those entities.The conclusion of the thesis was that Wikidata hierarchies largely did not correspond to the subject tag list on hand. The subject tag list and the sought-after hierarchies and relationships between them were too dissimilar to the general Wikidata ontology.

The methods used might have been more viable had the subject tag list been more general in nature. The graph structure proved to be a good way to maintain a hierarchical ontology, but the entity relationships had to be discovered via another source.

Keywords/tags (subjects)

BERT, Databases, Graph databases Gremlin, Machine learning, Neptune, Ontology, Python, SPARQL

Sisältö

Käsitteet....3
1 Johdanto....4
1.1 Toimeksiantaja ja tausta....4
1.2 Menetelnät ja tavoitteet....4
1.3 Tutkimusasetelma....5
2 Lähdedata....6
2.1 Sanasto....6
2.2 Wikidata....6
2.2.1 Esittely....6
2.2.2 Entiteettimappaukset....7
2.2.3 Wikidatan tietorakenteet....9
2.2.4 SPARQL ja kyselyt....12
2.3 Hierarkian hyödyntäminen....12
3 Toteutus....15
3.1 Neptune....15
3.3.1 Esittely....15
3.3.2 Graafin hyödyt....15
3.3.2 Gremlin....16
3.2 Jupyter....16
3.3 Graafin luonti....17
3.3 Graafiin yhdistäminen....21
3.4 Hierarkian haasteita....24
4 Lopputulos....26
4.1 Graafitaulu....26
4.2 Jatkokehitysmahdollisuudet....33
5 Pohdinta....34
Lähteet....36

Kuviot

Kuvio 1. Esimerkki wikidatan rajapinnasta palautuneesta entiteetistä....**7**
Kuvio 2. Kyselyfunktio wikidatan REST-apia varten....**8**
Kuvio 3. Esimerkki Wikidatan triploista graafimuodossa....**9**
Kuvio 4. Wikidatan kyselyrajoitukset....**10**

Kuvio 5. Esimerkki Wikidata-entiteetistä.....**11**
Kuvio 6. Esimerkki SPARQL-kyselystä.....**12**
Kuvio 7. Esimerkki pyramidimaisesta hierarkiarakenteesta.....**13**
Kuvio 8. Esimerkki monesta yläluokasta.....**14**
Kuvio 9. Esimerkkikulku Greminillä.....**16**
Kuvio 10. Graafikannan luonti AWS:n käyttöliittymässä.....**17**
Kuvio 11. Graafikannan luonti AWS:n käyttöliittymässä.....**18**
Kuvio 12. Tiedon lisääminen JSON-tiedostosta.....**19**
Kuvio 13. Yksinkertainen Gremlin-haku Pythonilla.....**19**
Kuvio 14. Tätä ajetaan etsimään yhteyksiä asiasanojen välillä.....**20**
Kuvio 15. Jos ID2 ei ole graafissa, etsitään sen ylätasot.....**21**
Kuvio 16. Haetaan entiteeteille kuvaukset wikidatasta.....**21**
Kuvio 17. Lambda-funktion oletusasetukset.....**22**
Kuvio 18. Lambda-funktion parametrit.....**23**
Kuvio 19. Lambdan käyttäminen Pythonilla.....**24**
Kuvio 20. Graafiin tallennetut asiasanat.....**26**
Kuvio 21. Asiasanat, joilla on Wikidata-entiteetti.....**27**
Kuvio 22. Esimerkkejä asiasanoista, joille ei löytynyt wikidata-entiteettiä.....**27**
Kuvio 23. Haetaan lista kaarista.....**28**
Kuvio 24. Muutamia wikidatasta haettuja kuvauksia.....**29**
Kuvio 25. Kuvaaja graafiin muodostuneista entiteettisuhteista.....**30**
Kuvio 26. Järkeviä, mutta orpoja yhteyksiä.....**31**
Kuvio 27. Suurin rypäs toisiinsa liittyviä asiasanoja graafissa.....**32**
Kuvio 28. Entiteettejä, joille löytyi useita ylätasoja.....**33**

Käsitteet

Amazon Neptune on Amazonin tuottama graafikanta.

Asiasana tai vaihtoehtoisesti **tagi** on joku asia, joka voidaan luokitella tekstin aiheeksi. Tämmöisiä asiasanoja voi käyttää luokitteluun tai hakusanoina.

BERT (Bidirectional Encoder Representations from Transformers) on erittäin yleinen algoritmi luonnollisen kielen tunnistuksen tarpeisiin ja työtä tehdään osaksi sitä varten, että BERT-malleja pystyisi kouluttamaan paremmin. (Devlin, J. Chang, M-W. Lee, K. Toutanova, K. 2019)

Graafitietokanta on tietokantarakenne, joka esittää tietoa solmujen (node) välisistä suhteista. Graafi koostuu vertekseistä (vertex) ja niitä yhdistävistä kaarista (edge). (What is a Graph Database? n.d)

Gremlin ja **SPARQL** ovat graafitaulujen (kuten Neptune tai Wikidata) käsittelyyn ja hakemiseen käytettäviä kyselykieliä.

Jupyter on notebook-ohjelmointiin käytetty alusta, joka on erityisen yleinen data-analyysin parissa.

Luonnollisen Kielen Käsittely (Natural Language Processing, NLP) on koneälytutkimuksen alahaara, jonka tavoitteena on rakentaa algoritmeja ja malleja, jotka pystyvät käsittelemään ja luokittelemaan ihmisten tuottamaan (siis luonnollista) tekstiä. (What Is Natural Language Processing? 2020)

Malli tässä raportissa viittaa koneoppimismalliin, mikä on jollakin datalla koulutettu algoritmi, joka pyrkii luokittelemaan sille annettua uutta materiaalia niihin luokkiin, jotka siihen on koulutettu. Neuwo.ai:n mallien käyttämät luokat ovat pääosin tämän työn käsittelemiä asiasanoja.

Ontologia tarkoittaa yleensä asioiden syvällisen olemuksen filosofista tutkimista, mutta tässä raportissa sillä tarkoitetaan tietopankkia asiasanoista, jossa asiasanoista myös tiedetään jotain. (Hofweber, 2017)

Python on korkean tason ohjelmointikieli, joka on yleisessä käytössä koneälytutkimuksessa. Tämä työn ohjelmointiosat on kirjoitettu Pythonilla.

1 Johdanto

1.1 Toimeksiantaja ja tausta

Tämän opinnäytetyön toimeksiantaja oli Media M1 Oy, joka on vuonna 2000 perustettu mainostoimisto. Media M1:llä on noin 40 työntekijää Jyväskylässä ja Helsingissä ja tuottaa pääosin mainoskampanjoita yrityksille. Opinnäytetyö liittyy Media M1:en Neuwo.ai-sivuhankkeeseen, joka tuottaa sisällönluokittelua, branditurvallisuustietoa ja muuta metadataa sanomalehdille. Pääosin Neuwo.ai tuottaa näitä palveluita tarjoamalla rajapintoja koneoppimismalleihin.

Aihe valittiin, koska ontologian rikastamisesta pystyi erottamaan riittävän laajan kokonaisuuden, jota kuitenkin pystyi tekemään pääosin itsenäisesti. Työssä pystyi myös käyttämään pääosin julkisia resursseja, joten ei ollut tarvetta viitata suuremmin Media M1:n liikesalaisuuksiin. Aihe sivuaa myös omia kiinnostuksenkohteita tietotekniikan alalla, sillä työn tavoitteena on loppujen lopuksi vahvistaa ja rikastaa koneälyn ja koneoppimismallien käyttöä.

1.2 Menetelmät ja tavoitteet

Opinnäytetyön tavoitteena oli rakentaa ontologiaverkko valmiiksi olemassa olevalle sanalistalle, jota Neuwo.ai:n käyttämät koneoppimismallit käyttävät. Sanalistan käytössä oli ollut haasteita, sillä sanalista oli rakennettu keräämällä Neuwo.ai:n käytössä olleesta koulutusmateriaalista löytynyt asiasanoitus tietämättä mitkä asiasanat liittyivät toisiinsa ja mitkä olivat tosiasiassa alakategoriota toisista asiasanoista. Koska tätä tietoa ei käytössämme olleessa datassa ollut, oli tarpeen rakentaa tämä tieto itse. Litteästä listasta tuli rakentaa verkko, johon sisältyy tieto mitkä asiasanat liittyvät toisiin asiasanoihin ja mitkä ovat asiasanojen yläkategoriat.

Työhön käytettyihin menetelmiin kuului Python-ohjelmointi, jolla haettiin tietoa erityisesti wikidatan rajapinnoista ja käsiteltiin saatua tietoa. Lisäksi tehtiin jonkin verran manuualista tietojen arviointia. Tämän lisäksi käytettiin SPARQL- ja Gremlin kyselykieliä graafitaulujen käsittelyyn ja lopputulos hostattiin Neuwo.ai:n aws-pilvessä. Ontologialle suunniteltiin graafirakenne, johon tiedot voitiin tallentaa.

Työtä tehtiin AWS:n infrastruktuurin sisällä, jotta tuloksia pystytiin helposti käyttämään Neuwo.ai:n pilvessä. Yhteensopivuus AWS:n pilven kanssa ja AWS:n tarjoama tuki oli merkittävä tekijä teknologioiden valinnassa.

1.3 Tutkimusasetelma

Tämän työn lähtöasetelmassa lähdettiin etsimään Neuwo.ai:n sanaston dataan vastaavuuksia wikidatasta, jotta ne pystyttäisiin paremmin luokittelemaan. Koska ongelmana oli tiedon puute ja tiedon siirtäminen oikeaan paikkaan, oli hankala tietää etukäteen, mikä oli järkevin paikka etsiä tietoa. Wikidata on valittu alustavaksi lähteeksi, mutta työssä pyritään myös selvittämään saako tämän tiedon haettua wikidatasta järkevästi. Työn alussa on myös tehty olettamus että asiasanoille löytyy wikidata-entiteetit. Näin oletettiin, koska Wikidata-entiteettejä on miljoonia ja asiasanoja on vain tuhansia. Tämä tulee kuitenkin empiirisesti vahvistaa.

Tietotekniikan ja myös koneoppimisen alalla pyritään usein välttämään asioiden tekemistä käsin ja tämäkin tyon lähtöasetelmassa oletetaan, että sanaston entiteettien suhteet voi päätellä ja määritellä ohjelmallisesti ja käsityön voi täten minimoida. Usein kuitenkin käsin luokittelemalla saa aikaan parempaa dataa, jota ei tarvitse jälkikäteen korjata. Tässä työssä tutkitaan myös wikidatasta kerättyjen tulosten laadullisuutta ja sitä tuleeko niitä täydentää tai jopa korvata käsin annotoimalla.

Sanastosta halutaan tietää asiasanojen väliset suhteet ja niiden yläkategoriat. Työn lähtöasetelmassa ei olla varmoja pystyykö tämän tiedon päättelemään suoraan lopullisesta graafista, vai joutuuko tämänkin asian määrittelemään erikseen.

Tutkimuskysymykset ovat siis seuraavat:

- Löytyykö jokaisella asiasanalle vastine Wikidatan tietorakenteista?
- Mille osalle niistä sellainen löytyy?
- Mitä tehdään silloin, kun asiasanalle ei löydy vastinetta?
- Saako Wikidatan hierarkiat suoraan nostettua kokonaan tai osittain Neuwo.ai:lle hierarkiaksi?
- Missä määrin hierarkiaa täytyy täydentää muilla lähteillä ja omien arvioiden perusteella?

2 Lähdedata

Tähän opinnäytetyöhön käytetty data tulee Neuwo.ai:n keräämästä koulutusmateriaalista ja wikidatasta löytyvistä entiteettikuvauksista. Neuwo.ai:n koulutusmateriaali koostuu pääosin annotoiduista uutisteksteistä.

2.1 Sanasto

Työn pohjana oli sanasto, johon Neuwo.ai:n asiasanoitusmallit perustuivat. Sanasto oli kerätty Neuwo.ai:n hankkimista asiasanoitetuista artikkeleista, joita käytettiin koulutusmateriaalina asiasanoitusmalleille. Sanastoa oli tämän jälkeen sekä supistettu että laajennettu poistamalla käytöstä sellaisia asiasanoja, joille ei löytynyt riittävästi näyttöä, että niitä voisi kouluttaa BERT-malliin ja lisäämällä asiasanoja, jotka olivat erityisen tarpeellisia ja joille pystyttiin tuottamaan koulutusmateriaalia tai ennustamaan muin keinoin. Käytössä oleva sanasto muuttui myös tätä opinnäytetyötä tehdessä. Tässä opinnäytetyössä käytettyyn sanastoon sisältyi yli 9000 asiasanaa.

Sanastoon kuului monenlaisia asiasanoja. Jotkut olivat paikkakuntia, julkisuuden henkilöitä, tai järjestöjä. Toiset ovat asiakokonaisuuksia kuten esimerkiksi ”Laki, Oikeus ja Rikos”. Sanastoa oli alunperin koottu suomeksi, mutta sitä on sittemmin käännetty ruotsiksi ja englanniksi tosin varsinkin ruotsinkieliset käännökset olivat pääosin konekäännöksiä.

Työn aloitusvaiheessa sanaston asiasanoilla oli kolme erikielistä versiota ja URI, jolla asiasanat pystyttiin yksilöimään, vaikka kirjoitusasu tai jopa asiasanan varsinainen merkkijono muuttuivat.

2.2 Wikidata

2.2.1 Esittely

Ontologiatietojen pääasiasiallisena lähteenä oli Wikidata. Wikidata on ilmainen ja julkinen tietograafi, jonka entiteetit on luokiteltu graafirakenteeseen. Wikidatassa siis on haluamamme kaltainen ontologia ja tietorakenne, joten on järkevää yrittää peilata wikidatasta löytyvää tietoa Neuwo.ai:n sanastoon. Wikidatasta löytyy lähes sata miljoonaa entiteettiä paljon laajemmasta listasta käsitteitä kuin Neuwo.ai:n sanasto. On siis todennäköistä, että enemmistölle käsitteistä on ainakin joku wikidatan entiteetti, jonka voidaan katsoa olevan sama asia kuin jokin asiasana.

Wikidataa voidaan käyttää heidän REST-rajapinnan kautta. Tällä keinolla pystymme löytämään wikidata entiteettejä, jotka vastaavat asiasanoja ja löytämään niiden yksilöivät tunnukset Wikidatan järjestelmässä. Varsinaisen tietorakenteen ja wikidatan graafin tutkimiseen ja hierarkiatietojen löytämiseen tarvitaan Wikidatan SPARQL-kyselypalvelua.

Wikidata ei ole ainoa olemassa oleva ontologia, jossa on hierarkia. Esimerkiksi Suomen Kansalliskirjasto ylläpitää Finto-ontologiaa. YAGO (Yet Another Great Ontology) on toinen avoin ontologia, jota olisi mahdollisesti voinut käyttää samaan tarkoitukseen. Tähän työhön valittiin kuitenkin Wikidata, sillä sen entiteettikokoelma on laajin ja sinne lisätään asioita aktiivisesti.

2.2.2 Entiteettimappaukset

Ennen kuin Wikidatasta löytyviä hierarkioita pystyi käyttämään Neuwo.ai:n sanaston hierarkioiden löytämiseen, sanaston asiasanoille täytyi löytää vastine Wikidatan tietokannoista. Joissain tapauksissa tämä oli helppoa, sillä tekemällä asiasanalla kyselyn wikidatan rajapintaan sai vastauksena takaisin vain yhden wikidata-entiteetin. Näissä tapauksissa oli kohtuullista olettaa, että löydetty entiteetti oli kohtuullinen kuvaus löydetystä asiasanoista. Hankalammissa tapauksissa mahdollisia entiteettejä oli useita tai ei yhtään. Jos entiteettejä löytyi useita, jouduttiin käsin käymään läpi löydetyt vastaukset ja valitsemaan niistä oikea. Niissä tapauksissa, joissa entiteettiä ei löytynyt wikidatasta hakemalla kyseinen asiasana jätettiin ilman yhteyksiä muihin asiasanoihin ontologian ensimmäisessä versiossa. Alla on esimerkki wikidata-entiteetistä ”suomen jalkapallo”, joka löytyy myös asiasanana Neuwo.ai:n sanastossa. (Ks. kuvio 1.)

{"id": "Q2740753",
"title": "Q2740753",
"pageid": 2632194,
"repository": "wikidata",
"url": "//www.wikidata.org/wiki/Q2740753",
"concepturi": "http://www.wikidata.org/entity/Q2740753",
"label": "football in Finland",
"description": "association football practiced in Finland",
"match": {"type": "alias", "language": "en",
"text": "Finnish football"},
"aliases": ["Finnish football"]}

Kuvio 1. Esimerkki wikidatan rajapinnasta palautuneesta entiteetistä.

Yllä on esimerkki rajapinnasta saadusta wikidata-entiteetissä, joita etsittiin alla esitetyllä koodilla. Ohjelmalla (Ks. kuvio 2.) pystyy etsimään wikidata entiteettejä antamalla funktiolle termejä, joilla se suorittaa wikidata-haun. Tätä ajettiin isolle osalle Neuwo.ai:n sanastoa. Entiteeteille haettiin id ja otsikko (title), jotka tässä tapauksessa ovat sama Wikidatan yksilöivä Qnumero (Qnumber). Uri ja concepturi ovat viittauksia verkko-osoitteisiin, josta tämän entiteetin voi nähdä. Lisäksi vastauksesta löytyy jokin label eli selkokielinen otsikko ja siitä löytyy kuvaus.

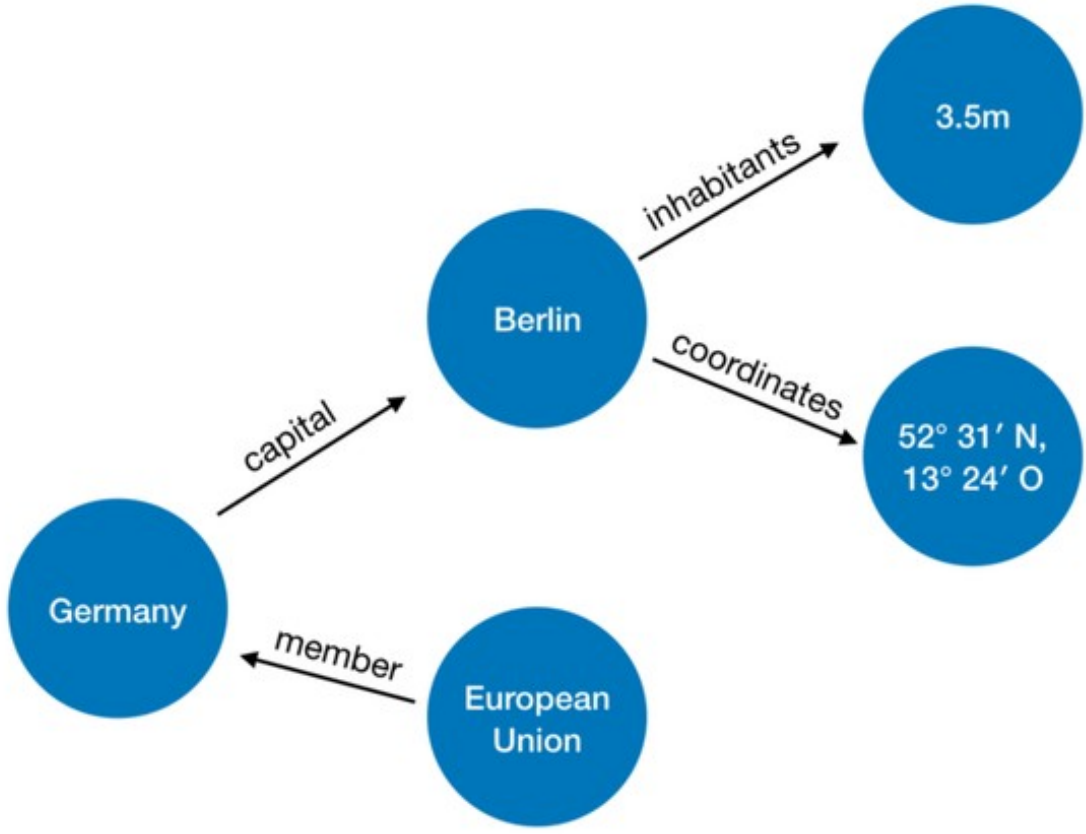
```
class WikiEntity(BaseModel):
    title: str
    pageid: str
    url: str
    concepturi: Optional[str]
    label: str
    description: Optional[str]


def get_entity(name: str) -> Optional[WikiEntity]:
    params = {
        'action': 'wbsearchentities',
        'format': 'json',
        'language': 'en',
        'search': name
    }
    try:
        response = requests.get("https://www.wikidata.org/w/api.php", params =
params)
        data = response.json()['search']
        return (data)
```

Kuvio 2. Kyselyfunktio wikidatan REST-apia varten.

Mappauksessa keskeinen tavoite on löytää jokaiselle käsiteltävän sanaston asiasanalle wikidatan Q-numero. Tällöin sanaston entiteeteillä on oman yksilöivän URI:n lisäksi viittaus wikidatan yksilöivään tunnukseen.

2.2.3 Wikidatan tietorakenteet

Wikidatan entiteetit yksilöidään Q-numeron (Qnumber) avulla ja ne koostuvat joukosta lauseita (statement) kyseisestä entiteetistä. (Wikidata: Help Statements, n.d.) Lause wikidatan formaatissa on jonkinlainen tripla, jossa on subjekti, predikaatti ja objekti. (Ks. Kuvio 3.) Hartmannin (2018) esittelyssä käytetään esimerkkinä hakua, jossa etsitään Euroopan unionin jäseniä. (Hartmann, 2018) Tällä haulla voitaisiin löytää esimerkiksi entiteetti Suomi (subjekti), jolla on jäsenyys (predikaatti) Euroopan Unioniin (objekti). Kaikki wikidatassa olevat tiedot sen eri entiteeteistä noudattavat tätä formaattia ja sitä voidaan käyttää SPARQL-kyselyjen rakentamiseen. Wikidataan voisi tällöin esimerkiksi tehdä kyselyn, jos etsitään entiteetille Suomi sen jäsenyydet, jolloin voisimme silloin päätellä, että Suomi ja Euroopan Unioni liittyvät toisiinsa. Jos molemmat ovat Neuwo.ai:n sanastossa asiasanoja, tämän saman tiedon voi kopioida graafikantaan kaareksi Suomea ja Euroopan Unionia kuvaavien asiasanojen verteksien välille. Wikidatan triplassa sekä subjekti että objekti ovat aina jonkinlaisia wikidata entiteettejä. (Ogbuji, 2018)



Kuvio 3. Esimerkki Wikidatan triploista graafimuodossa. (Hartmann, 2018)

Wikidatasta voi myös ladata datadumppeja, jotka noudattavat RDF-formattia. Datadumpin sisältö on pääosin samanlaista kuin mitä SPARQL-rajapinnasta pystyy löytämään ja itse tiedot ovat samat. Formaatissa ja esimerkiksi kenttien nimissä voi kuitenkin olla pieniä eroja. Tässä työssä käytetään kuitenkin rajapintaa, sillä datan määrä ei ole niin suuri etteikö sitä pystyisi kyselemään rajapinnasta ja dumpin käsittely toisi työhön uuden tarpeettoman komplikaation. Rajapintaa käytettäessä tulee kuitenkin tiedostaa, että Wikidata on julkinen resurssi, joka rajoittaa kyselymääriä yhdestä IP:stä suhteellisen aggressiivisesti, jos niitä lähettää liian usein. (Ks. Kuvio 4.) Jopa kolmenkymmenen sekunnin viiveellä kyselyjen välissä SPARQL-rajapintaan törmättiin välillä kyselyrajoituksiin. (Malyshev, S. Lederrey, G. 2022)

```
<head>
<meta http-equiv="Content-Type" content="text/html;charset=utf-8
"/>
<title>Error 429 Too Many Requests - Please retry in 15 seconds.
</title>
</head>
<body><h2>HTTP ERROR 429</h2>
<p>Problem accessing /bigdata/namespace/wdq/sparql. Reason:
<pre>    Too Many Requests - Please retry in 15 seconds.</pre></
p><hr><a href="http://eclipse.org/jetty">Powered by Jetty:// 9.4
.12.v20180830</a><hr/>

</body>
</html>
```

Kuvio 4. Wikidatan kyselyrajoitukset.

Kuten dokumentaatiossa kuvaillaan, Wikidatan lauseita voi olla monenlaisia. Alla olevalla esimerkillä on lauseet ”instance of”, ”country”, ”located in the administrative territorial entity” ja puoli tusinaa muuta. (Ks. Kuvio 5.) Jonkinlaista hierarkiaa voisi lähteä miettimään monen eri lauseen pohjalta. Jos tätä esimerkkiä olisi laittamassa hierarkiaan voisi katsoa sen olevan vaikkapa hostellin alla, Brasilian alla tai molempien alla. (Wikidata: Help Statements, n.d.)

Kuvio 5. Esimerkki Wikidata-entiteetistä.

Ei ole kuitenkaan järkevää käydä läpi kaikkia mahdollisia Wikidata-lauseita, sillä monet niistä eivät edusta minkäänlaista hierarkiaa. Esimerkiksi tieto jonkin sijainnin koordinaiteista ei ole hyödyllinen asiasanoituksen ylätasojen määrittämiseen.

Tässä työssä kiinnostavimmat lauseet ovat entiteettien InstanceOf- ja SubclassOf-lauseet, sillä niillä voidaan parhaiten rakentaa haluttua hierarkiaa. SubclassOf-lause todennäköisesti sisältää esimerkiksi sellaista tietoa, että ”Suomalaiset Jalkapallojoukkkueet” ovat osajoukko ”Jalkapallojoukkueista”, mikä käy järkeen. InstanceOf näkyy myös ylemmässä esimerkissä ja on erityisen hyödyllinen silloin, kun puhutaan nimetyistä paikoista tai ihmisistä. ”Che Lagarto Hostel Porto de Galinhas” on instanssi entiteetistä ”hostelli” ja tämä on järkevä ylätaso.

2.2.4 SPARQL ja kyselyt

SPARQL on kyselykieli, jota voi käyttää tiedon hakemiseen graafikannoista ja wikidata tarjoaa erillisen rajapinnan SPARQL-kyselyjen tekemiseen. Wikidata-hakuja voi tehdä Wikidatan ylläpitämän käyttöliittymän kautta tai lähettämällä kyselyitä rajapintaan suoraan. Tässä työssä Wikidataa on käytetty rajapinnan kautta Python-skripteillä. Rajapintaa käytettäessä SPARQL-kysely laitetaan url-parametriksi. (A Gentle Introduction to the Wikidata Query Service, n.d)

Jotta voi tehdä SPARQL-kyselyn jostain entiteetistä, tarvitaan sen yksilöivä Q-numero (Qnumber). Alla näkyy esimerkki SPARQL-kyselystä, jossa etsitään teleskoopin yläluokat. (Ks. kuvio 6.)

```
SELECT ?item ?itemLabel ?itemDescription ?subclassof ?subclassofLabel ?
instanceof ?instanceofLabel
WHERE
{
  VALUES ?item {
    wd:Q296433 #find information about this specific q-value, in this case
"telescope"
    }
  OPTIONAL {?item wdt:P279 ?subclassof . } #find all classes that telescope (or
any other item listed above) is a subclass of.
  OPTIONAL {?item wdt:P31 ?instanceof . } #find all entities that telescope (or
any other item listed above) is an instance of.
  SERVICE wikibase:label { bd:serviceParam wikibase:language "en". }
}
```

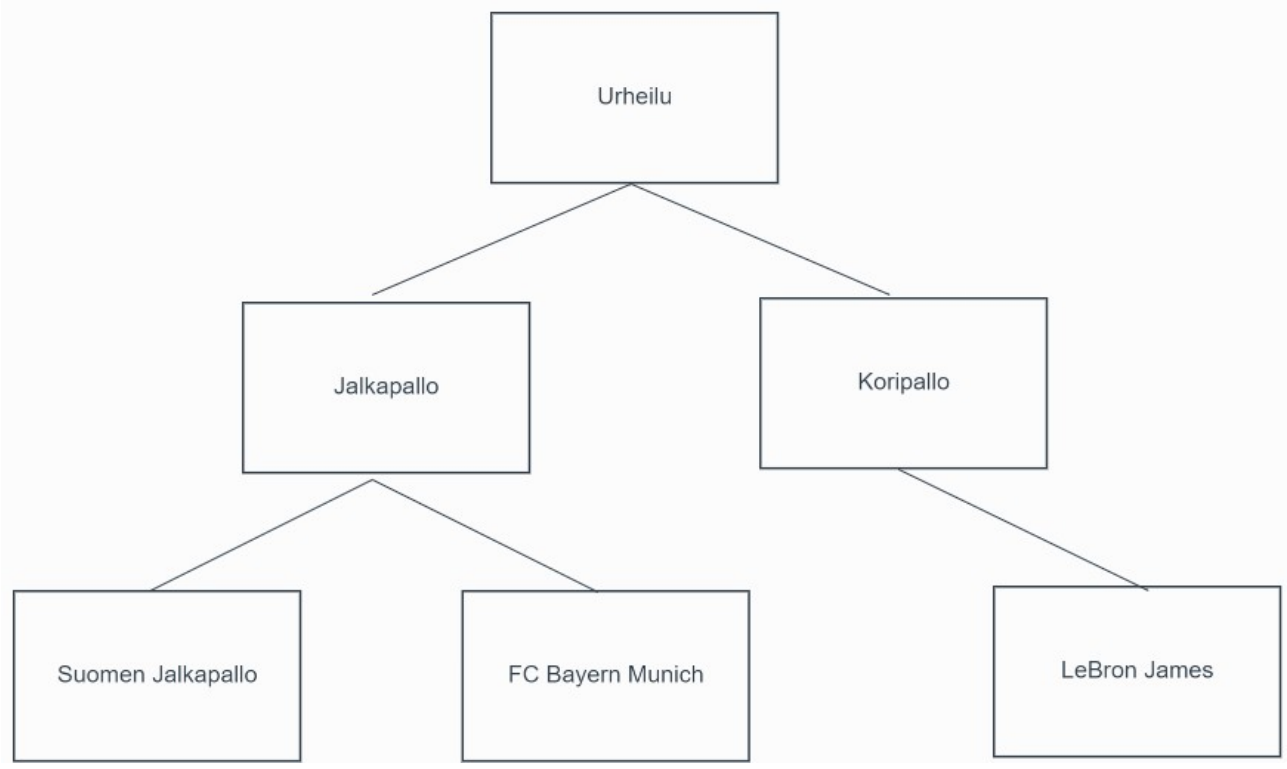
Kuvio 6. Esimerkki SPARQL-kyselystä.

2.3 Hierarkian hyödyntäminen

Hierarkiaa, sikäli kun sen pystyy olemassa oleville asiasanoille määrittämään, voidaan hyödyntää mallinnuksessa monin tavoin. BERT-malleissa suurempi määrä ennustettavia luokkia yleensä hankaloittaa koulutusdatan tuottamista. Keskimäärin mitä enemmän luokkia on yhden mallin ennustettavana, sitä hankalampaa on saada koulutusdataa jossa eri luokat ovat selkeästi erossa toisistaan ja sitä todennäköisemmin mallissa on ”meluista” dataa, jota on hankala hallita. Jos pystyy toteamaan, että jotkin olemassa olevat asiasanat joita nyt ennustetaan BERT-mallilla ovat toisten asiasanojen yläkategorioita, ei niitä tarvitse ennustaa ollenkaan. Nämä yläkategoriat

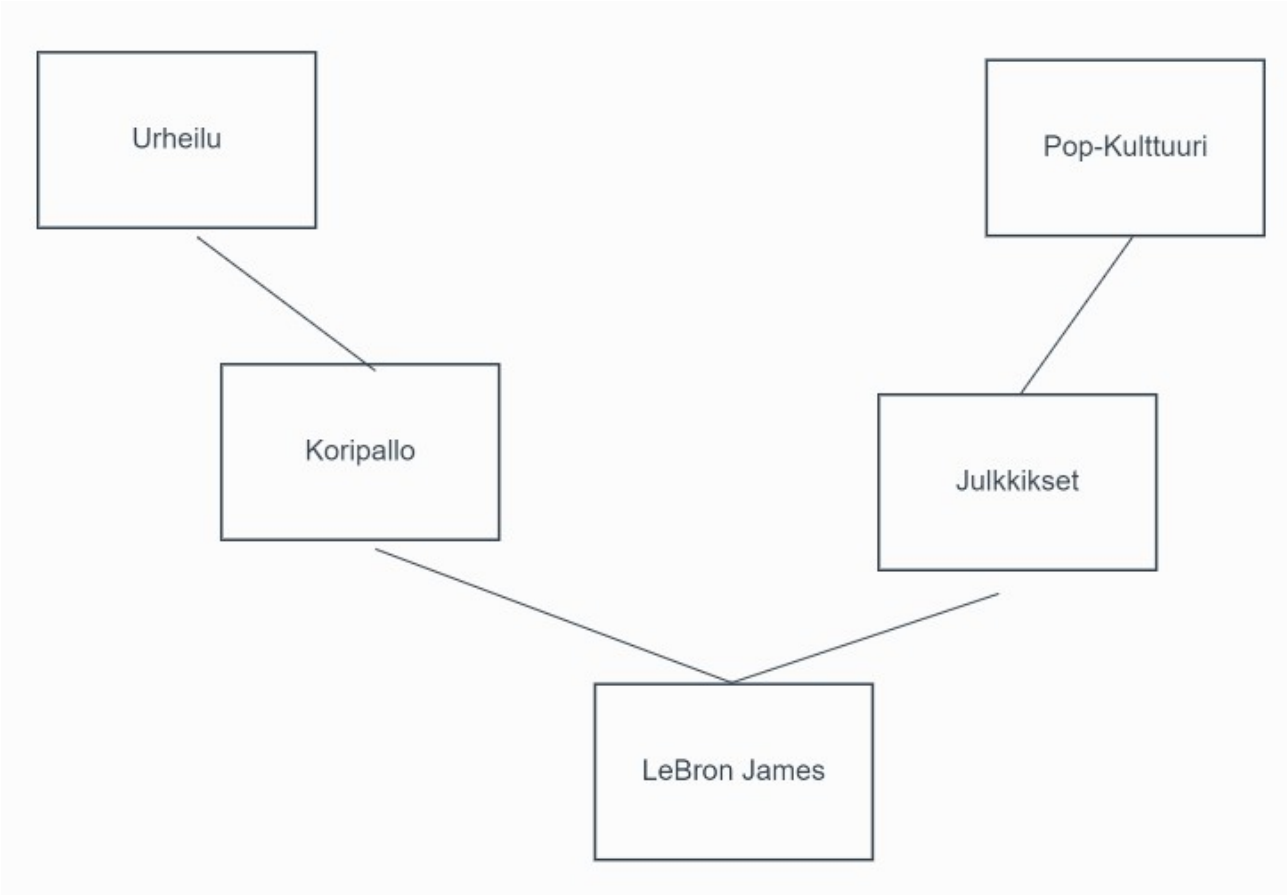
voidaan yksinkertaisesti päätellä mallinnuksen jälkeen ennustamalla alakategorioita. Jos hierarkia on rakennettu oikein, näin voidaan yksinkertaistaa mallia vähentämällä ennustettavia luokkia ja täten parantaa sen ennustuksia.

Toinen käyttötarkoitus oli mainonnan kohdentamisen tukeminen. Asiasanoja on tuhansia ja mainonnan kohdentamista ennustettujen asiasanojen perusteella jouduttiin usein määrittelemään käsin. Litteän listan kanssa tämä on hankalaa, koska relevantit asiasanat pitää käytännössä tapauskohtaisesti arvata ja ilman pientä ihmisen käsiteltävää listaa yläluokkia jotain jää helposti huomaamatta. Rakentamalla jokseenkin pyramidimainen hierarkia on mahdollista helpottaa asiasanaston käsittelyä. Alla on esimerkki mahdollisesta kolmikerroksisesta hierarkiarakenteessa, jossa ylätäso on ”urheilu” ja sen alla on välikategorioina erilaisia urheilulajeja. (Ks. Kuvio 7.) Mitään muita paitsi alimman tason asiasanoja ei välttämättä edes tarvitse yrittää malleilla ennustaa, sillä hierarkiasta voidaan päätellä että jos ”FC Bayern Munich” sopii asiasanaksi, niin sopivat myös ”jalkapallo” ja ”urheilu”.



Kuvio 7. Esimerkki pyramidimaisesta hierarkiarakenteesta.

Tässä vaiheessa ei tehty ratkaisua siitä tuleeko hierarkioiden olla jatkuvasti kapenevia. Esimerkiksi ylemmässä esimerkissä asiasana ”LeBron James” voitaisiin myös laittaa toiseen yläkategoriaan ”julkkikset”, joka lopulta löytyisi viimeisen yläkategorian ”pop-kulttuuri” alta. Tämänkaltaisia rakenteita ei varsinaisesti yritetä luoda, mutta graafimainen rakenne tekee niistä mahdollisia ja ne saattavat olla joillekin asiasanoille järkeviä. Alla on esimerkki mahdollisesti monen yläluokan rakenteesta. (Ks. Kuvio 8.)



Kuvio 8. Esimerkki monesta yläluokasta.

3 Toteutus

3.1 Neptune

3.1.1 Esittely

Amazon Neptune on nimensä mukaisesti Amazon Web Servicesin tuottama graafitietokanta, joka istuu natiivisti AWS:n pilvessä. Tämä tekee siitä hyödyllisen Neuwo.ai:n tarkoituksiin, koska muun järjestelmän käyttämä pilvi-infrastruktuuri tukee jo valmiiksi yhteyksiä Neptune-graafiin. Neptunea voi käyttää sekä SPARQL- että Gremlin kyselykielillä. Tässä työssä Neptune-hakuihin on käytetty Gremliniä, koska Gremlinin käyttöön tarjottu tuki ja dokumentaatio on laajempaa. (What Is Amazon Neptune? n.d.)

3.1.2 Graafin hyödyt

Kun tavoitteena on selvittää eri entiteettien suhteita toisiinsa, linkkejä eri käsitteiden välillä voi syntyä moneen suuntaan ja monilla eri tasoilla. Graafitaulu koostuu vertekseistä ja kaarista ja ei ole tarpeellista tietää millainen kaari on ja mitä se kuvastaa ennen sen lisäämistä. Täten entiteettisuhteiden tallentaminen graafiin on helpompaa. Lisäksi graafikantojen haut on optimoitu kaarten seuraamista varten ja ontologiaa tutkiessa ja käyttäessä kiinnostavimmat kysymykset ovat yleensä se minkä laajemman termin alle tämä termi kuuluu, mitä termejä tämän alle kuuluu ja mahdollisesta mitä muita samanlaisia termejä on olemassa. Graafitaulussa tämmöisen kyselyn voi tehdä seuraamalla yhtä kaarta lähtösolmusta. (What is a Graph Database? n.d)

Tämänkaltaisen tiedon tallentaminen perinteiseen relaatiokantaan on periaatteessa mahdollista, mutta se vaatii entiteettien suhteiden ennalta määrittelemistä ja on paljon jäykempää. Monisuuntaiset suhteet voivat olla hankalia rakentaa ja käyttää relaatiotietokannassa. Työtä aloittaessa ei ollut tiedossa voiko tietyllä entiteetillä olla esimerkiksi useita ylätasoja. Jos tietyllä asiasanalla voi olla sekä monta isäntää, että monta lasta, relaatiorakenteen ylläpitämisestä tulee vaikeaa tai jopa mahdotonta. Vaikka rakenne pidettäisiin täysin puumaisena, sen kyseleminen käytännössä luultavasti vaatisi jonkinlaisen erillisen ID-rakenteen jossa ID:stä pystyy suoraan päättelemään minkä asiasanan lapsista on kyse.

3.1.3 Gremlin

Gremlin on kyselykieli graffikannoille, jota tässä työssä käytetään lopullisen Neptune-graafin luomiseen ja kyselemiseen. Se sopii minkä tahansa graafikannan käyttämiseen, ku SQL sopii relaatiokantoihin.

Gremlinin ymmärtämiseen merkittävin käsite on ”kulkeminen” (traversal). Gremlin kyselyt yleensä alkavat jostakin solmusta, jonka jälkeen ne kulkevat jotain kaarta pitkin. Kulkuja voi ketjuttaa niin kauan kun löydetyistä solmuista löytyy ulospäin suuntautuvia kaaria, kunnes päätetään lopettaa ja palauttaa jotkut kiinnostavat arvot lopullisista löydetyistä solmuista. Alla oleva haku (Ks. kuvio 9.) etsii solmun jolla on nimi ”Gremlin” ja etsii sen tuntemat solmut ja sitten sen tuntemat solmut, joista se palauttaa nimet. (Gremlin Query Language, n.d)

```
g.V().has("name","gremlin").
  out("knows").
  out("knows").
  values("name")
```

Kuvio 9. Esimerkkikulku Greminillä. (Gremlin Query Language, n.d)

3.2 Jupyter

Työn visualisoitiin käytetään Jupyteriä AWS:n Sagemaker notebook-järjestelmän kautta. Jupyter tukee magic-lauseita, joilla voi yhdellä rivillä suorittaa operaatioita jotka muuten saattaisivat vaatia useita funktioita (Konkiewicz 2018). Tässä työssä käytetyin magic-lause on ”%%gremlin”, sillä se abstraktoi pois tarpeen käyttää funktioita graafikantaan yhdistämiseen.

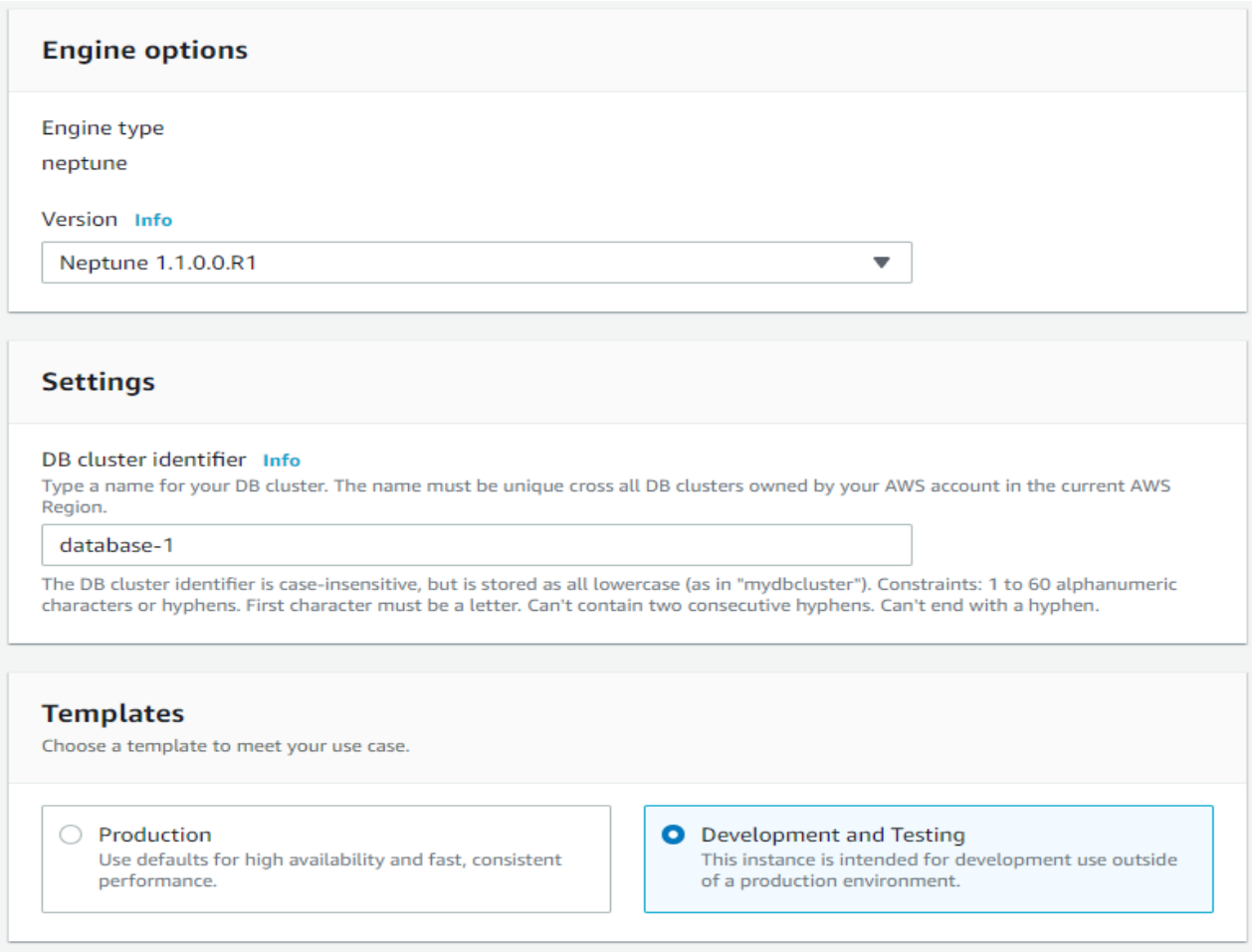
Jupyter-notebookin merkittävin etu on sen helppokäyttöisyys. Graafien, taulukoiden tai muun datan esittäminen visuaalisessa muodossa ei vaadi suurta määrää koodia. Abstraktion määrä tekee Jupyteristä sopimattoman raskaampiin ohjelmointitehtäviin, mutta erinoimaisen visualisaation tarkoituksiin. (Gutierrez 2020.)

3.3 Graafin Luonti

Graafi luotiin AWS:ään Neuwo.ai:n yksityisen virtuaaliseen pilveen. Neptunea ei voi käyttää AWS:n infrastruktuurin ulkopuolella, joten jos tämän työn haluaisi hostata itse tai tehdä AWS:n pilven ulkopuolella, pitäisi valita jokin toinen graafitietokanta. Alla näkyy mistä voi luoda uuden Neptune-kannan AWS:n käyttöliittymässä. (Ks. Kuvio 10.) (What Is Amazon Neptune? n.d.)

Kuvio 10. Graafikannan luonti AWS:n käyttöliittymässä.

Alustavasti graafille valittiin oletusasetukset. Koska testaus ja kehitysvaiheessa ei haluta käyttää kaikkein järeimpiä virtuaalikoneita, valitaan ”Development and Testing” konfiguraatio. Tämä käynnistää suhteellisen kevyen db.t3.medium luokan tietokantainstanssin jolla on 4 gigabittiä keskusmuistia. Tämä paketti ei myöskään sisällä peilejä useille saatavuusalueille. Siinä vaiheessa kun graafin nopeus ei ole kriittinen tekijä ja taulua ei käytetä reaaliajassa, ei tarvita suurempaa kapasiteettia. Kun tuloksia aletaan myöhemmin soveltaa Neuwo.ai:n käyttöliittymään ja muuten käyttämään suoraan asiasanojen hallintaa, luotettavuuteen ja kapasiteettiin panostaminen voi olla tarpeellista. (Ks. Kuvio 11.)



Kuvio 11. Graafikannan luonti AWS:n käyttöliittymässä.

Kehitysvaiheessa graafikannalle valittiin suhteellisen kevyt db.t3.medium-tietokantavirtuaalikone, jota ei ole peilattu useille AWS:n saatavuusalueille. Kanta laitettiin samalle Frankfurtin alueelle kuin muut Neuwo.ai:n resurssit, jotta sitä voisi helposti käyttää muista AWS:ssä hostatuista Neuwo.ai:n virtuaalikoneista. (What Is Amazon Neptune? n.d)

Graafi alustettiin ottamalla JSON-tiedostoja, joihin oli tallennettu asiasanojen URI, suomenkielinen, ruotsinkielinen ja englanninkielinen nimi, sekä wikidatan yksilöivä Q-numero. JSON-tiedosto käytiin läpi ja lisättiin graafitauluun käyttämällä alla olevaa koodia. Näin saadaan graafitauluun solmut, joille voidaan sitten lisätä kaaria kuvaamaan entiteettien välisiä suhteita. (Ks. kuvio 12.)

```
statics.load_statics(globals())

graph = Graph()

remoteConn = DriverRemoteConnection(NeptuneURL,'g')
g = graph.traversal().withRemote(remoteConn)

data = json.load(json_file)
for tag in data:
    tagentry = {}
    URI = tag["URI"]
    fin = tag["fin"]
    eng = tag["eng"]
    Qnumber = tag["candi"]["id"]
    g.addV('tag').property("id", URI).property("finnish",
fin).property("english", eng).property("Qnumber", Qnumber).next()

remoteConn.close()
```

Kuvio 12. Tiedon lisääminen JSON-tiedostosta.

Kun graafissa on solmuja, niille pitää löytää joitain kaaria, jotta voidaan puhua hyödyllisestä graafista. Alla olevalla ohjelmalla palautuu kaikkien graafissa olevien solmujen Q-numerot käytössä olevasta Neptune-taulusta. (Ks. kuvio 13.)

```
from gremlin_python import statics
from gremlin_python.structure.graph import Graph
from gremlin_python.process.graph_traversal import __
from gremlin_python.process.strategies import *
from gremlin_python.driver.driver_remote_connection import
DriverRemoteConnection
import requests
import json

statics.load_statics(globals())

graph = Graph()

remoteConn = DriverRemoteConnection(NeptuneURL,'g')
g = graph.traversal().withRemote(remoteConn)
ID= g.V().has('Qnumber').values('Qnumber').toList()
print(ID)
remoteConn.close()
```

Kuvio 13. Yksinkertainen Gremlin-haku Pythonilla.

Suurin osa hierarkian hauista tehdään alla olevalla tavalla. Graafin tallennettujen solmujen ID:t otetaan talteen ja niille lähdetään etsimään ylätasoja. Ensimmäinen askel oli yksinkertaisesti käyttää wikidatan SPARQL-rajapintaa suoraan ja etsiä jokaisen asiasanan ensimmäiset ylätasot, jonka jälkeen voidaan katsoa löytyvätkö nämä ylätasot sanastosta. Tämä yksinkertainen lähestymistapa ei kuitenkaan tuottanut kovin paljon tuloksia, sillä suurin osa wikidatan ylätasoista ei ole Neuwo.ai:n asiasanoissa. Alla olevassa ohjelmassa etsitään taulusta jokainen solmu, jolla on Q-numero ja niille tehdään hakuja wikidatan SPARQL-kyselypalveluun. (Ks. kuvio 14.) Ohjelmassa haun url sisältää SPARQL-kyselyn, joka hakee taulusta löydetylle Q-numerolle InstanceOf- ja SubclassOf tietoja.

```
response = requests.request("POST", url, headers=headers, data=payload,
files=files)
    data = response.json()['results']
    print("data found")
    items = data["bindings"]
    for i in items:
        Q = i["instanceof"]["value"]
    SubQnumber = Q.strip("http://www.wikidata.org/entity/")
    ID1 = g.V().has('Qnumber', Qid).next()

    ID2 = g.V().has('Qnumber', SubQnumber).next()

    edge = g.V(ID1).as_('t').V(ID2).addE("subclassof").to("t").toList()
```

Kuvio 14. Tätä ajetaan etsimään yhteyksiä asiasanojen välillä.

Tätä lähestymistapaa laajennettiin lisäämällä toinen kierros ylätasojen hakemiseen. Mikäli tietyn asiasanan ylätasot eivät ole sanastossa, etsitään ylätasojen ylätasot ja tarkistetaan ovatko ne sanastossa. (Ks. kuvio 15.) Tällä lähestymistavalla löydettiin jonkin verran lisää suhteita. Kaikille asiasanoille ei kuitenkaan löytynyt sanastosta ylätasoja.

```
ID2 = g.V().has('Qnumber', SubQnumber).next()
        if not ID2:
                response = requests.request("POST", url, headers=headers,
data=payload, files=files)
            data = response.json()['results']
            items = data["bindings"]
            for i in items:
               if i["instanceof"]:
                  Q = i["instanceof"]["value"]
               ThirdQnumber = Q.strip("http://www.wikidata.org/entity/")
               ID2 = g.V().has('Qnumber', ThirdQnumber).next()
               edge = g.V(ID1).as_('t').V(ID2).addE("subclassof").to("t").toList()
```

Kuvio 15. Jos ID2 ei ole graafissa, etsitään sen ylätasot.

Lopulta haettiin niille asiasanoille, joille löytyi wikidata-entiteetit jokin kuvaus wikidatasta. (Ks. Kuvio 16.) Tämän voi tehdä missä tahansa vaiheessa prosessia, sillä kuvauksen olemassaolo ei vaikuta mihinkään muuhun hakuun.

```
conn = create_remote_connection()
g = graph.traversal().withRemote(conn)

IDs = g.V().has('Qnumber').not_(has("description")).values('Qnumber').toList()
for i in IDs:
   entity = get_entity(i)
   if entity and entity.description
      description = entity.description
      ID = g.V().has("Qnumber", i).property("description", description).iterate()
```

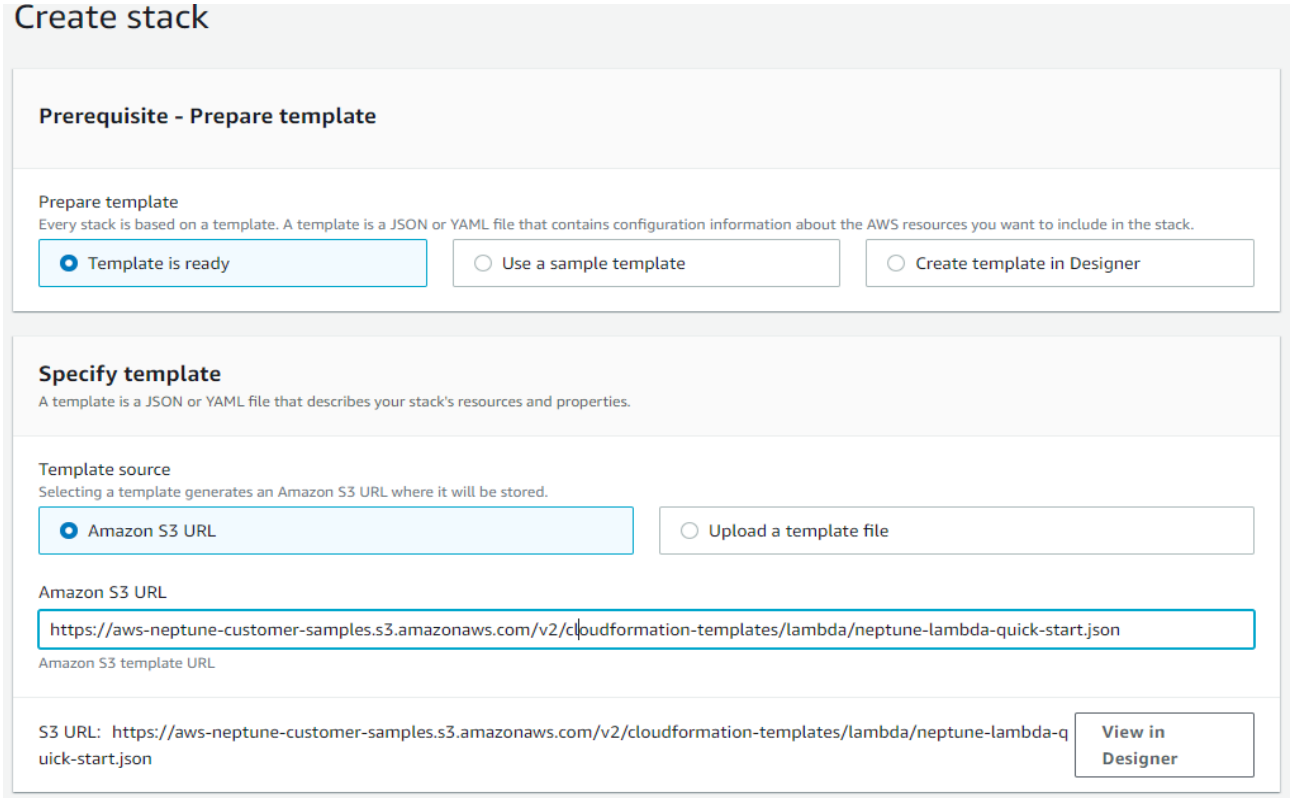
Kuvio 16. Haetaan entiteeteille kuvaukset wikidatasta.

3.4 Graafiin yhdistäminen

Neptune graafitaulun käyttäminen vaatii oikeuksien ja yhteyksien määrittämisen AWS:n ekosysteemissä. Alustavaa testikäyttöä graafikannalle luotuun käyttöoikeusryhmään (security group) lisättiin pääsy testaamiseen käytetystä EC2-virtuaalikoneesta. Yhden instanssin oikeudet on helppo laittaa pystyyn, siihen ei tarvita kuin EC2-instanssin osoitteen lisäämistä graafikannan käyttöoikeusryhmään. Tämä ei kuitenkaan ole kestävä ratkaisu, sillä graafia voidaan haluta tulevaisuudessa käyttää monista eri paikoista Neuwo.ai:n pilvessä. Voi esimerkiksi syntyä tarve

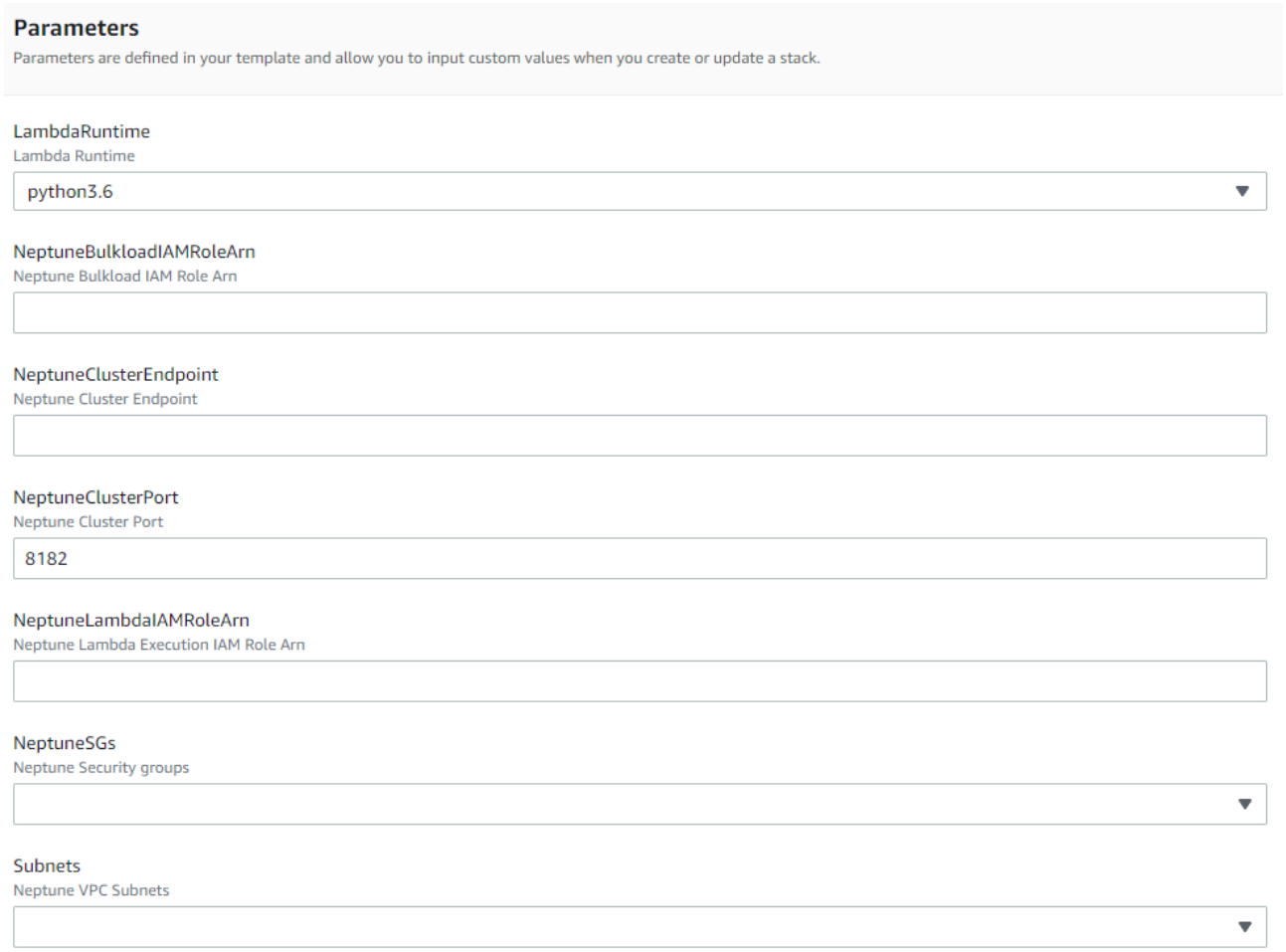
yhdistää graafi käyttöliittymäportaaliin tai hakea dataa Jupyter-notebook instanssin kautta S3-ämpäriin.

Tähän tarkoitukseen tulee tehdä lambda-funktio, jota voidaan käyttää siltana graafitauluun. Tässä tapauksessa voidaan käyttää hyvin yksinkertaista lambdaa oletusasetuksilla, sillä sen ei tarvitse tehdä mitään muuta, paitsi antaa muille AWS:n resursseille pääsy graafitauluun. Alla näkyy AWS-lambda funktion oletusasetukset. (Ks. Kuvio 17.)



Kuvio 17. Lambdan-funktion oletusasetukset.

Lambdalle tarvitaan porttinumero, neptune-endpoint, neptunen käyttöoikeusryhmä ja virtuaalipilven subnet. Tässäkin selvitään pääosin oletusasetuksilla. (Ks. Kuvio 18.)



Kuvio 18. Lambda-funktion parametrit.

Kun Lambda on luotu, tarvitaan koodit lambda-funktion käyttämiseen. (Ks. kuvio 19.) AWS suosittelee käyttämään ajastettuja tokeneita ja alla oleva ohjelma olettaa sellaisten käytön. Tällä rakenteella on mahdollisuus käyttää graafia ilman, että Neptune käyttöoikeusryhmään täytyy antaa erikseen oikeudet jostakin EC2-virtuaalikoneesta tai vastaavasta lähteestä. Huomattiin, että voidakseen käyttää graafia Sagemakerin notebook-instanssista, on tarpeen lisätä itse notebook Neptune-kannan käyttöoikeusryhmään.

```
def prepare_iamdb_request(database_url):
    service = 'neptune-db'
    method = 'GET'

    access_key = ""
    secret_key = ""”
    region = ""
    session_token = ""”

    creds = SimpleNamespace(
        access_key=access_key, secret_key=secret_key, token=session_token,
region=region,
    )

    request = AWSRequest(method=method, url=database_url, data=None)
    SigV4Auth(creds, service, region).add_auth(request)
    return httpclient.HTTPRequest(database_url,
headers=request.headers.items())


def create_remote_connection():
    print('Creating remote connection')

    return DriverRemoteConnection(connection_string(), 'g')


def connection_string():
    database_url = ('')
    return prepare_iamdb_request(database_url)

statics.load_statics(globals())
graph = Graph()

conn = create_remote_connection()
g = graph.traversal().withRemote(conn)
```

Kuvio 19. Lambdan käyttäminen Pythonilla. (Using AWS Lambda functions in Amazon Neptune, n.d)

3.5 Hierarkian haasteita

Haasteisiin törmättiin jo entiteettimappausvaiheessa, sillä kaikille asiasanoille ei löytynyt selkeää vastinetta Wikidatasta. Jos asiasanaan ei pystynyt yhdistämään Wikidata-entiteettiä, sille ei myöskään pystynyt peilaamaan Wikidasta hierarkiaa tai entiteettisuhteita. Tällaiset asiasanat voi silti lisätä graafiin, mutta niiden suhteet on löydettävä muualta. Kaikille entiteeteille ei myöskään löytynyt yläkategorioita sanastosta, vaikka niillä olisikin ollut wikidata-entiteetit mapattuina.

Wikidatasta saa siis vain osittaisen ontologian rakennettua suoraan tilanteessa, jossa asiasanat eivät alunperin olleet Wikidatan entiteettejä. Vaikka Wikidataan on tallennettu todella suuria määriä dataa, sinne ei kuitenkaan ole tallennettu kaikkea mahdollista.

Toinen haaste johon törmättiin oli se, että ontologiahierarkia ei ole yksiselitteinen käsite. Se mikä on järkevä kategorisaatioperiaate riippuu ontologian käyttötarkoituksesta ja Wikidata tai esimerkiksi Finto käyttävät hyvin yleismaailmallisia ontologioita joiden tarkoitus on neutraalisti kuvailla maailmaa. Neuwo.ai:ta ja sen asiakkaita sen sijaan kiinnostavat asiasanojen markkinointiarvo ja nämä kaksi lähestymistapaa eivät välttämättä sovi yhteen. Esimerkiksi Wikidatasta löytyy monista julkisuuden henkilöistä ensimmäisenä yläkategoriana ”ihminen”, mutta tämä on markkinointialustan asiasanoituksen tarpeisiin hyödyllinen tieto vain siitä näkökulmasta että henkilö, jolle löytyy wikidata-entiteetti, jossa on jotain tietoa on luultavasti jollain tasolla merkittävä henkilö. Se, että henkilö on ihminen ei tuo oikeastaan mitään markkinointiarvoa ja tämän asiasanoituksen hierarkian rakentamiseen olisi paljon hyödyllisempää esimerkiksi tunnistaa että ihminen on koomikko, jolloin kategoriaa ”koomikko” voisi käyttää mainonnan kohdentamiseen.

Näiden ongelmien ratkaiseminen käytännössä vaatii hierarkioiden ja yläkategorioiden käsin korjaamista ja tarkoittaa, että ratkaisua ei saada pelkästään käyttämällä olemassa olevaa ontologiadataa. Sitä on myös tuotettava itse.

Kysymykseksi heräsi myös mikä on järkevä määrä ala- ja ylätasoja. Koska yksi hierarkian käyttötarkoituksista on tuottaa johdonmukaisia luokkia joiden avulla voidaan kohdistaa mainontaa useaan eri asiasanaan, työssä on muitakin rajoituksia kuin pelkkä filosofinen tutkimus asiasanojen suhteista. Viittä suurempi määrä tasoja katsottiin liialliseksi ja kolme todettiin hyväksi kohdearvoksi. Siis hierarkian ylimpien tasojen ja sen alimpien välillä olisi normaalitilanteessa vain yksi väliluokka. (Rantanen, 2022)

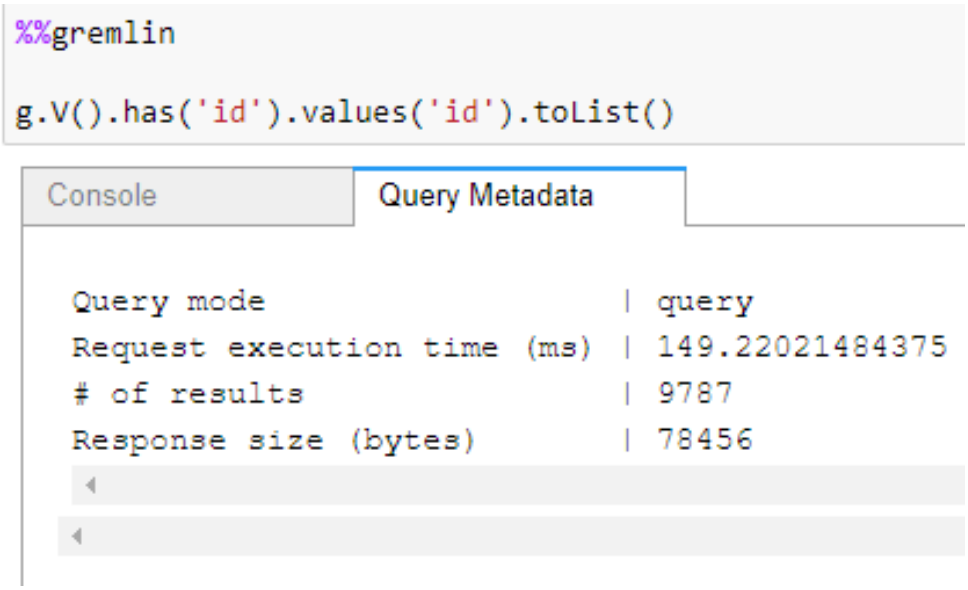
Tutkiessa Wikidatasta löydettävää tietoa todettiin myös, että ylätasot on ainakin jollain tasolla rajattava etukäteen. Wikidatan peilaamisesta ja edes Q-numeroiden etsimisestä löytyi riittävästi aukkoja, että oli tarpeellista tehdä jokin rajaus, jonka pohjalta tehdä manuualista lajittelua. Tällaista lajittelua ei ollut mahdollista tehdä ennen ennalta määriteltyjä luokkia, joten alettiin

määrittelemään ylätasoja asiasanoja varten. Näitä ylätasoja pystyi myös käyttämään Wikidatan tietojen varmentamiseen.

4 Lopputulos

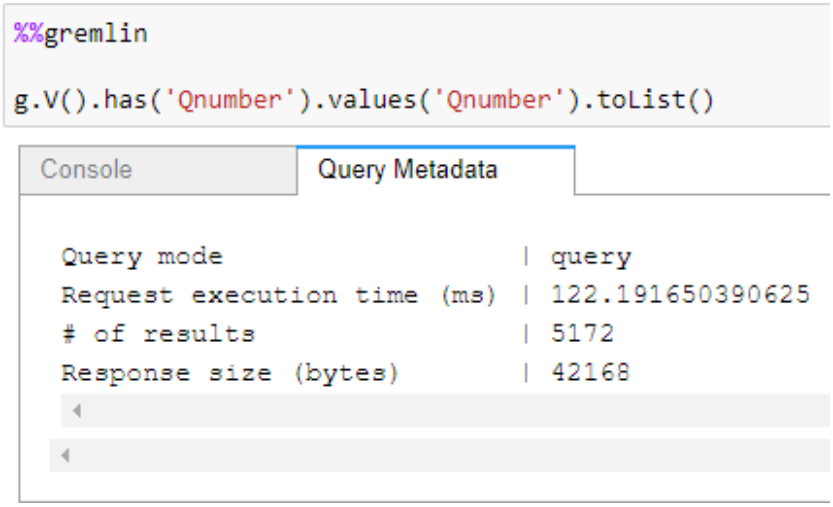
4.1 Graafitaulu

Lopullisen graafiin tallennettiin 9787 asiasanaa. (Ks. Kuvio 20.) Jokaisella asiasanalla on id-arvo, joka on alkuperäiseen sanastoon kuulunut URI.



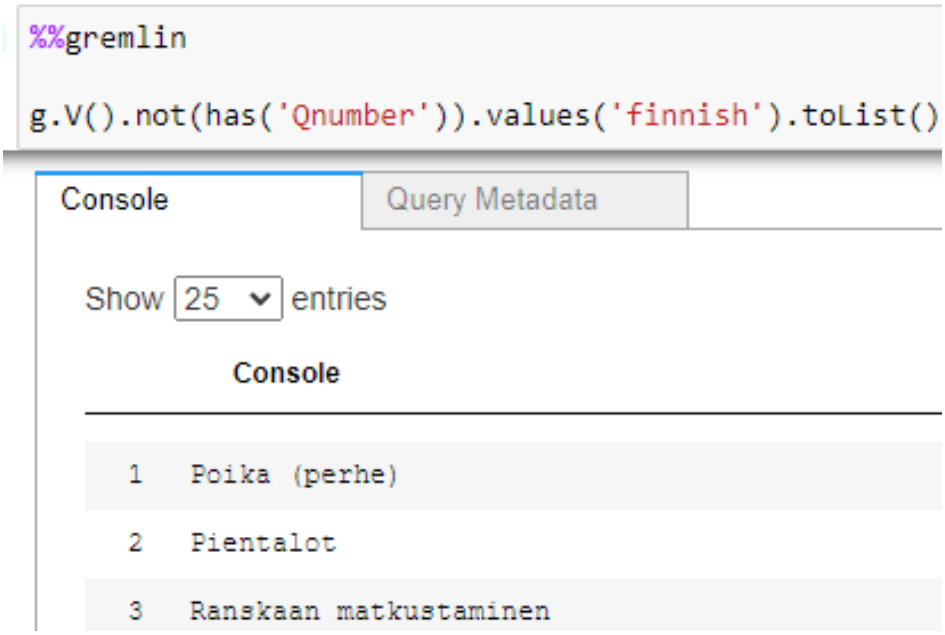
Kuvio 20. Graafiin tallennetut asiasanat.

Asiasanoista 5172:lla on jokin Wikidata-entiteetti. (Ks. Kuvio 21.) Osasta on pystytty vahvistamaan silmämääräisesti, että entiteetti vastaa todella asiasanaa, mutta suurimmassa osassa on käytetty ensimmäistä wikidatan rajapinnasta löytynyttä Q-numeroa. Tutkimusasetelmassa kysyttiin, kuinka suurelle osalle asiasnoista löytyy jokin vastine. Hiukan alle puolelle löytyi jotain.



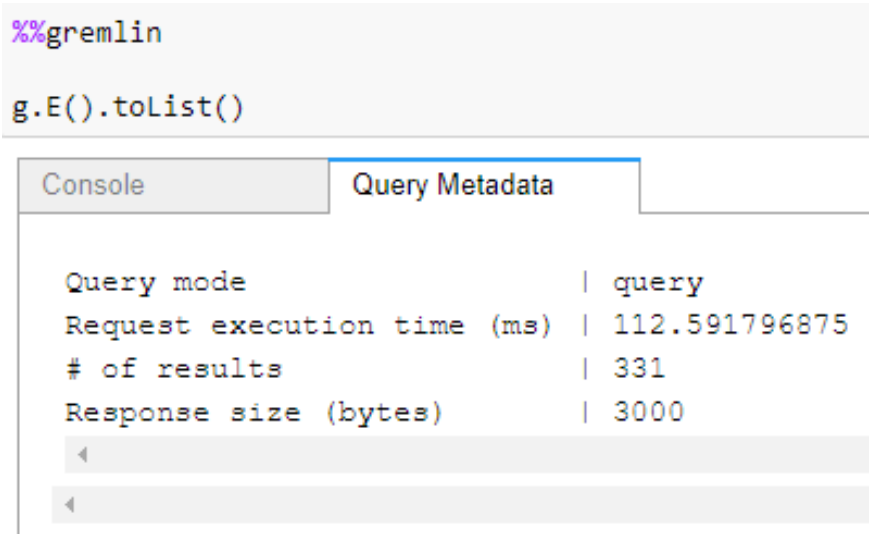
Kuvio 21. Asiasanat, joilla on Wikidata-entiteetti.

Voidaan siis todeta, että wikidata-entiteetti löytyi hieman yli puolelle asiasanoista. Tämä on huonompi tulos kuin lähtöasetelmassa odotettiin. Yksi syy tähän voi olla, että monet asiasanaston termit eivät ole niin konkreettisia kuin wikidatan entiteetit yleensä. Alla näkyy pieni otos asiasanoja, joille ei löytynyt vastaavia termejä. (Ks. Kuvio 22.) ”Poika (perhe)” saattaa johtua hakukoodien rajoituksista. Jonkinlainen entiteetti, joka merkitsee miespuolista lasta luultavasti löytyy, mutta sitä ei automaattisella haulla onnistuttu hakemaan. ”Pientalot” ja ”Ranskaan matkustaminen” voivat taas olla niin abstrakteja käsittejä, että hyvin konkretiaan pohjautuva wikidata ei niitä käsittele.



Kuvio 22. Esimerkkejä asiasanoista, joille ei löytynyt wikidata-entiteettiä.

Itse entiteettisuhteiden löytyminen oli kuitenkin vielä rajallisempaa. (Ks. Kuvio 23.) Pelkästään wikidatan pohjalta, asiasanastolle löytyi 331 kaarta. Tämä tarkoittaa että 5172:n entiteetin välillä, on vain 331 hierarkiasuhdetta. On siis erittäin epätodennäköistä, että millekään asiasanalle löytyy toimivaa hierarkiaa. Suuri enemmistö asiasanoista on edelleen täysin orpoja, eikä wikidataa käyttämällä niille onnistuttu löytämään mitään suhdetta toisiin asiasanoihin. Käytännössä Wikidatan hierarkiaa ei saanut nostettua Neuwo.ai:lle hierarkiakse missään merkittävässä kapasiteetissa.



Kuvio 23. Haetaan lista kaarista.

Wikidatasta onnistuttiin kuitenkin keräämään entiteetteille kuvauksia. Vaikka hierarkian rakentaminen wikidatan pohjalta epäonnistui, wikidataa pystyi silti käyttämään asiasanojen rikastamiseen sikäli kun entiteettimappauksia onnistui tekemään. (Ks. Kuvio 24.) Kuvaus saatiin suurimmalla osalle niistä asiasanoista, joilla oli wikidata-entiteetti.

```
%%gremlin

g.V().has('Qnumber').has("description").values('description').toList()
```

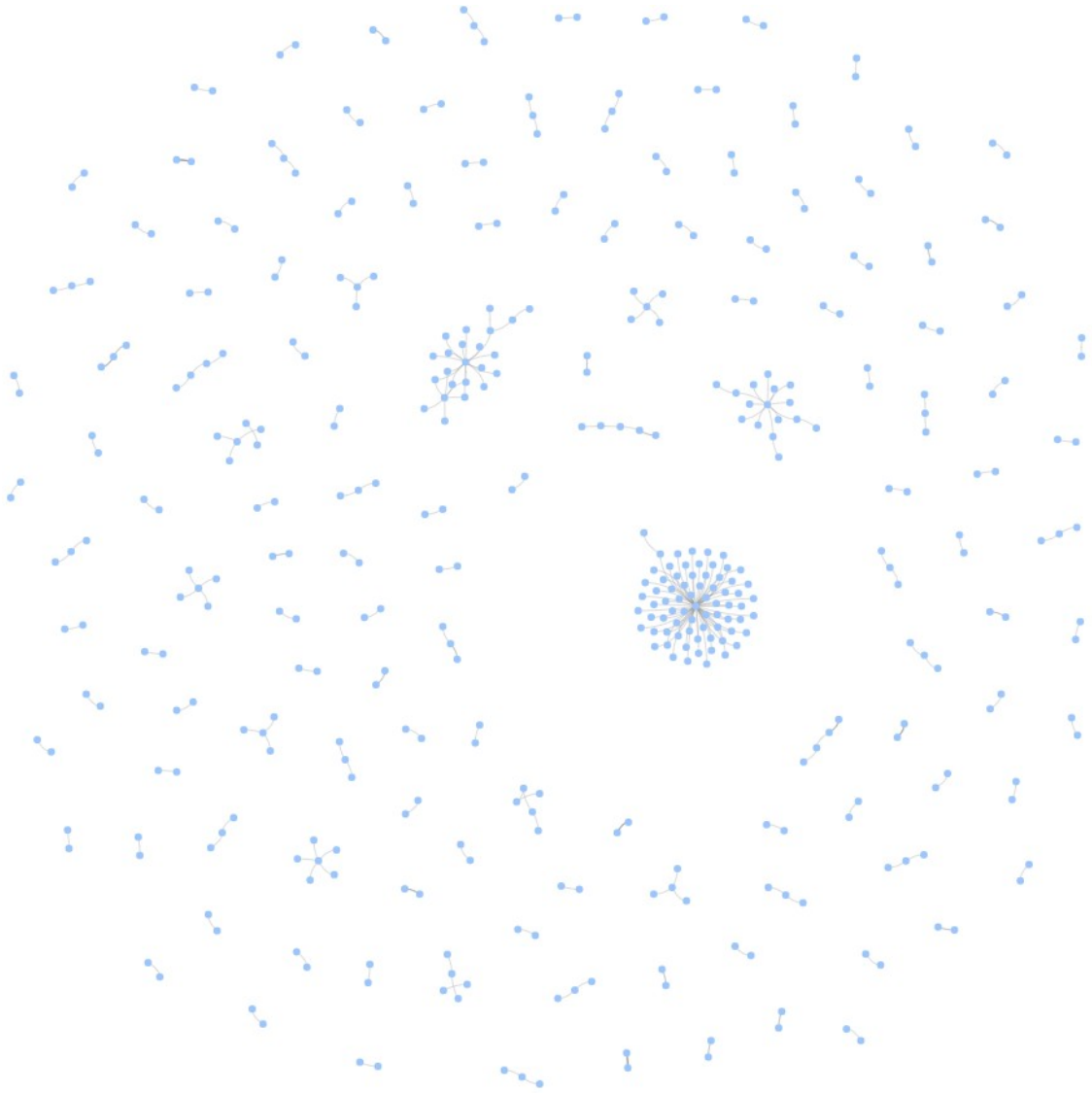
Console | Query Metadata

Show 25 entries Search:

	Console
1	city in the region of Kymenlaakso in Finland
2	country in northern Europe
3	maximum legal speed of vehicles
4	structure, typically with a roof and walls, standing more or less permanently in one place
5	activity of trying to catch fish
6	overview of music traditions in Finland
7	popular music genre

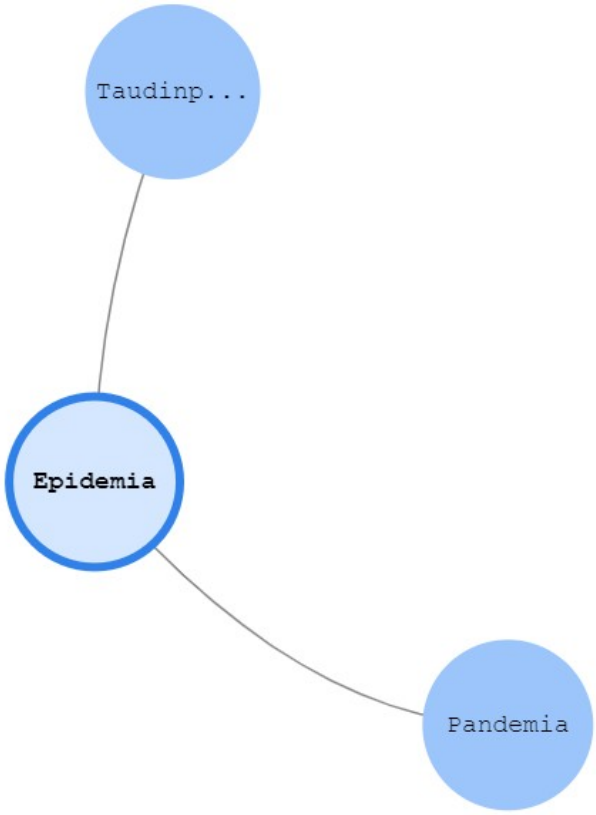
Kuvio 24. Muutamia wikidatasta haettuja kuvauksia.

Sikäli kun hierarkiaa löytyi, se muodostui irrallisiksi ryppäiksi. Tahtotila eli rakenne, jossa asiasanoista muodostuisi korkeintaan parikymmentä puuta ei toteutunut. Kuvaajasta puuttuvat myös se enemmistö asiasanoista, joilla ei ole mitään yhteyksiä muihin asiasanoihin. Graafiin on muodostunut pääosin muutaman asiasanan naruja, mutta siellä on myös muutama isompi rypäs tageja. (Ks. Kuvio 25.)



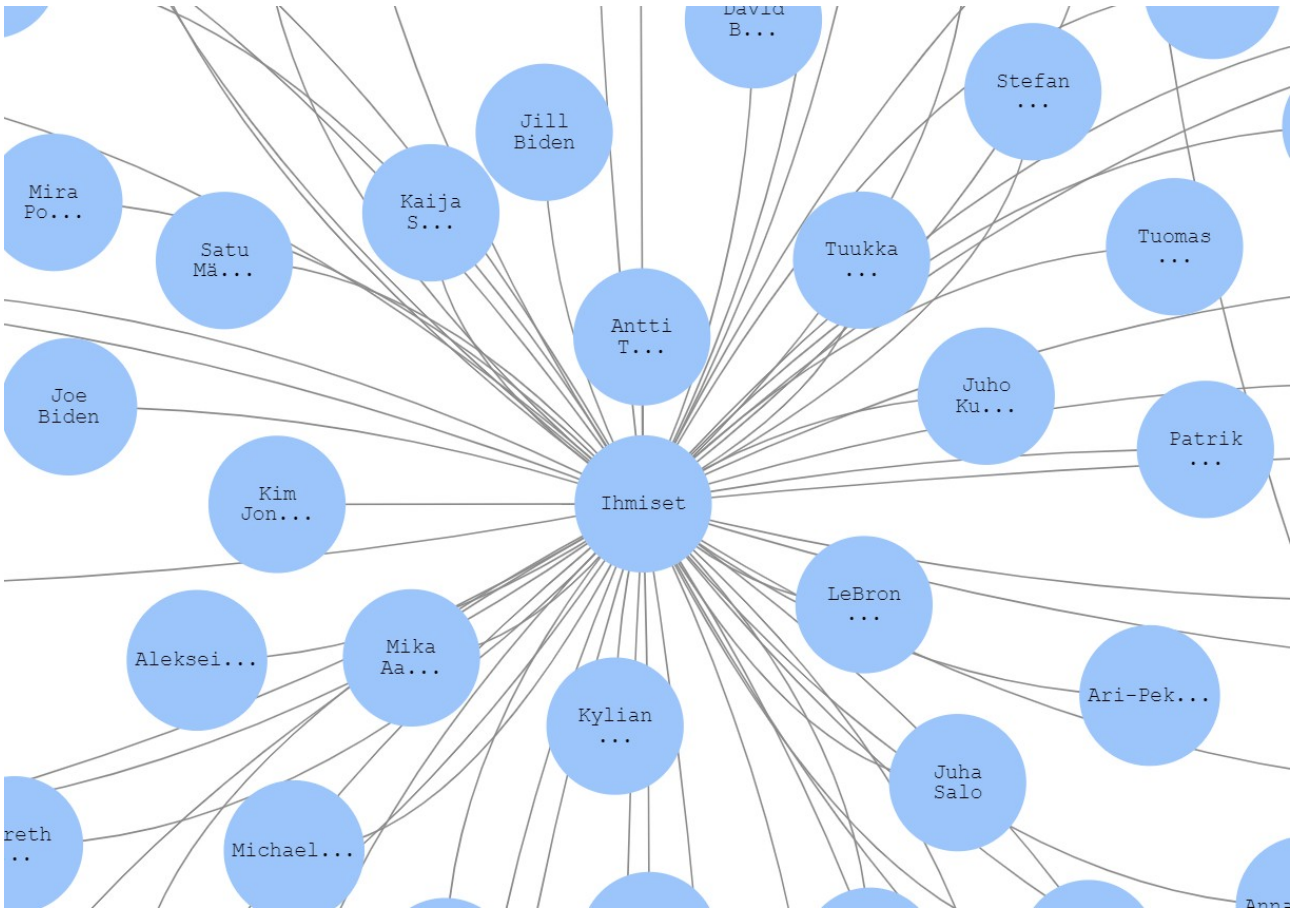
Kuvio 25. Kuvaaja graafiin muodostuneista entiteettisuhteista.

Jotkin löydetyistä naruista ovat ihan käyttökelpoisia. Esimerkiksi pandemia on selkeästi osajoukko epidemia-tagia, joka on itsessään osajoukko taudinpurkaus-tagia. (Ks. Kuvio 26.) Tämänkaltaisia yhteyksiä voitaisiin käyttää koulutusdatan parantamiseen. Jos johonkin tekstiin kuuluu asiasana ”pandemia”, voitaisiin tämän tiedon perusteella oikein päätellä, että siihen kuuluu myös ”epidemia” ja ”taudinpurkaus”.



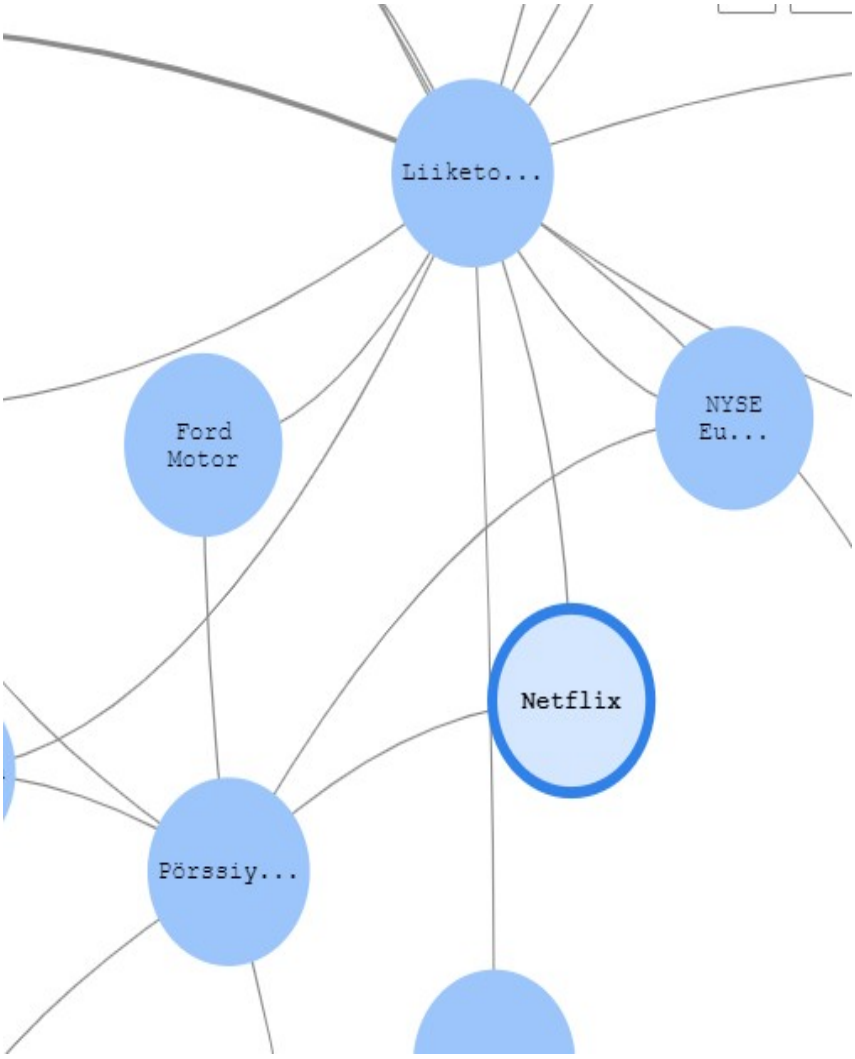
Kuvio 26. Järkeviä, mutta orpoja yhteyksiä.

Suuri rypäs keskellä kuvaajaa muodostuu ihmiset-asiasanasta, jolle tulee suuri joukko asiasanastosta löytyviä julkisuuden henkilöitä. (Ks. Kuvio 27.) Tämä on hyvä esimerkki wikidatan käytön rajoituksista, sillä vaikka on loogisesti järkevää laittaa julkisuuden henkilöt kategoriaan ihmiset, tämä ei ole Neuwo.ai:n mainoskohdennusontologiaa varten hyödyllistä tietoa.



Kuvio 27. Suurin rypäs toisiinsa liittyviä asiasanoja graafissa.

Graafista löytyi myös ainakin yksi tapaus jossa ylätasoja oli muodostunut kaksi. (Ks. Kuvio 28.) Ylätasot ”Liiketoiminta” ja ”Pörssiyhtiö” ovat molemmat saaneet alleen monia samoja yrityksiä, kuten ”Netflix” ja ”Ford Motor”. Molemmilla ylätasoilla on kuitenkin myös omia alatasoja, joita ei ole toisella. Esimerkiksi ”Liiketoiminta” asiasanan alla on ”Foreca”. ”Liiketoiminta” ja ”Pörssiyhtiö” eivät kuitenkaan ole suoraan yhteydessä toisiinsa.



Kuvio 28. Entiteettejä, joille löytyi useita ylätasoja.

4.2 Jatkokehitysmahdollisuudet

Työn alussa kysyttiin joutuuko hierarkiaa täydentää muista lähteistä ja mitä tehdään, jos asiasanoille ei löydy vastineita. Yksi vaihtoehto olisi yrittää löytää yhteyksiä esimerkiksi YAGO:sta tai Finton ontologiasta samalla tavalla kuten wikidatasta haettiin tietoa. Tässä lähestymistavassa on kuitenkin se riski, että näiden julkisten lähteiden data on yhtä vaillinnaista kuin wikidatasta saatu.

Toinen vaihtoehto on tuottaa dataa itse luokittelemalla. Tämä tarkoittaisi, että tulee määritellä jotkut ylätasot etukäteen. Graafi on kuitenkin hyvä ratkaisu tällaisen datan tallentamiseen, vaikka sitä joutuisikin tuottamaan käytännössä käsin.

Koska Neptune-taulu on jo integroitu AWS-pilveen ja lambda-funktio sen käyttöä varten on olemassa, graafitaulun käyttäminen muissa Neuwo.ai:n palveluissa ei ole erityisen vaativaa. Gremlin-kyselyitä voitaisiin tehdä lambdan kautta ja sieltä löydettyjä tietoja näyttää esimerkiksi Neuwo.ai:n käyttöliittymässä.

Itse taulu on tämän työn puitteissa tehty suhteellisen kevyessä tietokantainstanssissa ilman useita saatavuusalueita tai muita AWS:n työkaluja varmistaa, että palvelu on luotettavasti saatavilla. Kun ontologiaa aletaan käyttää tuotannossa, tulee olemaan tarpeellista siirtää tai uudelleenluoda taulu sellaiseen kantainstanssiin, joka on riittävän kestävä.

5 Pohdinta

Työn alussa oletettiin, että koska Wikidatassa on yli sata miljoonaa entiteettiä ja käsiteltävässä sanastossa on vain noin kahdeksan tuhatta, Wikidatasta löytyisi kaikki entiteetit. Näin ei kuitenkaan ollut. Merkittävälle osalle asiasanoja ei löytynyt selkeästi vastaavaa Wikidata-entiteettiä, mikä pahensi seuraavaa ongelmaa.

Koska hierarkiaa lähdettiin etsimään paljon laajemmasta kentästä kuin alkuperäinen sanasto, oli odotettavissa, että asiasanoille löytyy ylätasoja, jotka eivät kuulu Neuwo.ai:n sanastoon. Kun hakuohjelmia kokeiltiin noin ~1700 asiasanalla, vain noin sadalle löytyi ylätaso joka kuului Neuwo.ai:n sanastoon. Tämä oli odotettavissa ja tätä pystyttiin osittain paikkaamaan toimenpiteillä kuten ylätasojen seuraamista usean kerroksen verran, mutta tämä aiheutti silti hierarkiaan lisää reikiä.

Kokonaisuudessaan lähestymistapa ei tuonut niin hyviä tuloksia kuin olisi voinut toivoa. Luokittelutyössä harvoin on ilmaisia lounaita ja tämä osoittautui todeksi tässäkin tapauksessa. On hankalaa tietää täsmälleen miksi sanasto ei vastannut Wikidataa kovin läheisesti, mutta on mahdollista tehdä muutamia hypoteeseja aiheesta. Sanastossa oli jonkin verran käsitteitä, joilla oli lähinnä arvoa markkinoinnin koostamisen kannalta, mutta joita ei välttämättä nähtäisi merkityksellisinä yleisen ontologian näkökulmasta. Samoista syistä Wikidata ei aina tukenut sellaisia entiteettisuhteita, joita olisi voinut toivoa asiasanastosta löytyvän.

Jos haluaa luokitella jotain koneoppimismaailmassa luotettavasti, sen joutuu yleensä tekemään aluksi itse. Yksi työn tavoitteista siinä mielessä epäonnistui, sillä Wikidatan käyttäminen ei vapauttanut tarpeesta tehdä manuaalista luokittelutyötä. Muihin tavoitteisiin kuitenkin päästiin, sillä esimerkiksi Neptune osoittautui hyväksi työkaluksi ylläpitää hierarkiaa.

Tämänkaltainen työ saattaisi onnistua paremmin erilaisella sanastolla. Jos vastaavaa työtä tekisi yleismaailmallisemmalla sanastolla ja käytettävän sanaston entiteettien vaadittaisiin alusta alkaen olevan esimerkiksi Wikidata-entiteettejä, hierarkian peilaaminen onnistuisi paljon helpommin. Julkisesti saatavilla olevat ontologiat eivät ole hyödyttömiä, vaikka niiden soveltuvuus Neuwo.ai:n sanastoon oli rajallinen.

Lähteet

A Gentle Introduction to the Wikidata Query Service. N.d. Esittely Wikidata dokumentaatiossa. Viitattu 17.4.2022.
https://www.wikidata.org/wiki/Special:MyLanguage/Wikidata:SPARQL_query_service/A_gentle_introduction_to_the_Wikidata_Query_Service

Devlin, J., Chang., M-W. Lee., K & Toutanova, K. 2019. BERT: Pre-training of Deep Bidirectional Transformers for Language Understanding

Gremlin Query Language. N.d. Esittely Apache Tinkerpop dokumentaatiossa. Viitattu 17.4.2022. https://tinkerpop.apache.org/gremlin.html

Hartmann, B. 2018. A Brief Introduction to Wikidata. Artikkeli KDNuggets verkkojulkaisussa. Viitattu 16.4.2022. https://www.kdnuggets.com/2018/05/brief-introduction-wikidata.html

Hofweber, T. 2017. Logic and Ontology. Viitattu 16.4.2022.
https://plato.stanford.edu/entries/logic-ontology/

Konkiewicz, M. 2020. Top 8 Magic Commands in Jupyter Notebook. Artikkeli About Data Blog-blogissa. Viitattu 4.5.2022. https://www.aboutdatablog.com/post/top-8-magic-commands-in-jupyter-notebook

Malyshev, S & Lederrey, G. N.d. RDF Dump Format. Ohjeita Wikidata dokumentaatiossa. Viitattu 16.4.2022. https://www.mediawiki.org/wiki/Wikibase/Indexing/RDF_Dump_Format

Ogbuji, U. 2018. Introduction to Wikidata as a platform and data source. Artikkeli IBM verkkosivulla. Viitattu 8.5.2022. https://developer.ibm.com/articles/use-wikidata-in-ai-and-cognitive-applications-pt1/

Ogidan, B. 2018. Graph Databases, Why are they important. Artikkeli Medium.com verkkosivuilla. Viitattu 4.5.2022. https://medium.com/the-andela-way/graph-databases-why-are-they-important-c438e1a224ae

Rantanen, J, 2022. Viitattu 12.4.2022. Teams-keskustelu ontologiarakenteen rajoituksista.

Using AWS Lambda functions in Amazon Neptune. N.d. Ohjeita Amazon Web Services dokumentaatiossa. Viitattu 27.4.2022.
https://docs.aws.amazon.com/neptune/latest/userguide/lambda-functions.html

What is a Graph Database? N.d. Ohjeita Amazon Web Services dokumentaatiossa. Viitattu 17.4.2022. https://aws.amazon.com/nosql/graph/

What Is Amazon Neptune?. N.d. Ohjeita Amazon Web Services dokumentaatiossa. Viitattu 17.4.2022. https://docs.aws.amazon.com/neptune/latest/userguide/intro.html

What is Natural Language Processing? 2020. Artikkeli IBM verkkosivuilla. Viitattu 17.4.2022. https://www.ibm.com/cloud/learn/natural-language-processing

Why You Should be Using Jupyter Notebooks. 2020. Artikkeli Open Data Science-blogissa. Viitattu 5.4.2022 https://odsc.medium.com/why-you-should-be-using-jupyter-notebooks-ea2e568c59f2

Wikidata: Help Statements. N.d Ohjeita Wikidata dokumentaatiossa. Viitattu 18.4.2022. https://www.wikidata.org/wiki/Help:Statements