Detecting communities in a language co-occurrence network

Harry Bitten h.bitten@se18.qmul.ac.uk	Maria Cross m.s.cross@se18.qmul.ac.uk

***Abstract*—Language networks link languages that are likely to be co-spoken. They can aid the study of languages' influences on global knowledge transfer. In this work, we investigated communities within a language co-occurrence network derived from book translations, aiming to find local groups in which individual languages have the strongest influence. We found two major communities in the weighted network, using two different agglomerative approaches, suggesting that there are not distinct groups of languages and that information sharing operates across the whole network.**

Keywords—languages, translation, network, community detection

I. Introduction

Global language co-occurrence networks (GLCNs) link languages that are likely to be co-spoken. Representing language co-occurrence as a network allows inference about international information sharing and knowledge transfer [1]. This includes the diversity of information received by speakers of the language, the speed at which the information will be received, and the ability of native speakers to globally disseminate information.

Ronen *et al.* hypothesised that the position of a language in a GLCN can serve as a metric to measure the global influence of said language [1]. Considering three distinct GLCNs derived from online and offline sources, they found that the eigenvector centrality of the language in the network is correlated with other measures of language influence, including the gross domestic product (GDP) per capita, and the number of famous native speakers.

In this work, as opposed to the global measure of a language's influence as discussed above, we studied GLCN communities - groups of languages with tighter links between themselves than with other languages. We hypothesised that detecting communities within a GLCN could reveal local groups in which particular languages have the greatest influence. Our main research questions were (1) can communities be detected in a GLCN? And (2) are the network communities robust, or do they differ depending on the detection algorithm used?

In the subsequent section of this report the related work is discussed, considering both community detection algorithms and their applications in GLCNs. We then describe the GLCN used in this work (section III) and the experimental study (section IV). In section V, we present our results, and finally, in section VI, conclusions are drawn.

II. Related work

A. Community detection algorithms

Many advances have drawn out the significance of understanding complex networks in a wide range of domains. One feature of complex networks are communities, which are defined as groups of nodes that have a higher likelihood of being associated to one another than to other individual nodes. The goal of a community detection algorithm is to assemble comparative hubs in a system into networks, whilst expanding the dis-similitude between them [2]. Many strategies have been proposed, yet a considerable amount of them are not reasonable for huge scale systems since they have high multifaceted nature and utilize worldwide learning.

Let us begin with the formal foundation over which we will build up the exchange of networks. This is the thought of a basic graph which is compromised by a set of nodes (or vertices) along with a set of links (or edges) among them [3]. A graph may be directed (whereby edges indicate a one-way relationship) or undirected (edges indicate a two-way relationship, in that each edge can be traversed in both directions). Moreover, a graph may be two-fold, in which case any connection between two hubs either exists or does not exist. On the other hand, a diagram may be weighted, in which case any connection is furnished with a specific weight or esteem, which is commonly a (positive) number.

The purpose here is to display an early on and brief exchange of the formal idea of a network with regards to the hypothesis of complex networks (and informal community investigation), and to portray (for the most part by models) several of the numerous computational systems which are normally utilized for the recognition of networks in a diagram theoretic foundation. Here, we portray the idea of language co-occurrence within Global language co-occurrence networks (GLCNs) by using existing algorithms to discover community structures in graphs [1].

Several techniques have been proposed in the literature to address the issue of instinctively inferring an idea by clustering. In general, they can be assembled into two classes: the similarity-based strategies and the set-hypothetical methodologies [3]. The former are portrayed by the utilization of a similarity/separate measure to figure the pairwise similarity between vectors comparing to terms to choose on the off chance that they are semantically comparative and accordingly ought to be clustered or not. Conversely, set-hypothetical methodologies mostly request the articles as per the incorporation connection between their capabilities.

The similarity-based clustering strategies are additionally classified into agglomerative (bottom-up) and divisive (top-down). The agglomerative approaches start with the focuses as individual groups and, at each progression, consolidates the most comparative or nearest pair of clusters. This strategy requires a denotation of cluster similarity or separation. In contrast, divisive approaches start with one comprehensive cluster and, at each progression, split into groups until single groups of individual focuses remain. For this situation, we have to choose, at each progression the cluster to split [4]. In this paper, we compare two agglomerative approaches.

Walktrap, developed by Pons *et al.,* is an algorithm in graph theory, used to identify communities in large networks via random walks [5]. These random walks are then used to compute distances between nodes. Nodes are then assigned into groups with small intra and larger inter-community distances via bottom-up hierarchical clustering. It should be noted, of course, that this algorithm considers only one community per node, which in some cases can be an incorrect hypothesis.

Label propagation, or LPA, proposed by Raghavan *et al,* is a near-linear community detection solution that benefits from fairly simple implementation. The algorithm works as follows [6]:

1. A network is characterized as a graph G(V, E), where V is the full set of nodes and E is the full set of edges.
2. For node i(i ∈ V) , let L_i denote the label of i, and N(i) denote the set of its neighbors
3. At the start of the process, each node is assigned a unique label e.g. $L_i = i$.
4. These labels then propagate throughout the network, with each node updating its label at every iteration to the one shared by most of its neighbors,

$$L_i=argmax_l|N^l(i)| \qquad (1)$$

5. This process is repeated until each node has one of the most frequent labels of its neighbors, that is, none of the nodes need to change their label.
6. Finally, communities are constructed of nodes that share the same label

B. *Communities in language co-occurrence networks*

There is a paucity of literature describing communities in GLCNs. In one study using 110 different language editions of Wikipedia, language communities were identified using common co-editing behaviours [7]. In this work, similarly to the Walktrap method described above, a community detection algorithm was applied using random walks within the network to identify groups of languages with strong inter-community ties (the Infogap algorithm). 21 communities containing two or more languages were detected. The authors concluded that community formation was highly correlated with linguistic similarities due to a shared language heritage (i.e. the language family), and shared religion.

However, to our knowledge, communities have not yet be explored in other GLCNs, such as those derived from book translations.

III. Dataset

The book translations global languages network from the Massachusetts Institute of Technology (MIT) was used in this study [1]. The dataset is a directed network, derived from over 2.2 million book translations published worldwide from 1979 to 2011. The translation data was obtained from UNESCO's Index Translationum - an international index of printed book translations.

In the network, nodes represent individual languages and edges indicate a translation of a book from one language to another. The frequency of the translation in the dataset - the co-occurrence frequency - can be used to weight the edges.

In comparison to the UNESCO dataset, Ronan *et al.* limited the range of languages following an international standard (ISO 639-3) [8]. Also, when constructing the dataset, the authors included only edges with a co-occurrence frequency greater than six, and where the probability of the connection is larger than expected based on the prevalence of the two languages [1],

$$P(edge_{i,j})>P(node_i)P(node_j). \qquad (2)$$

To calculate the strength of a connection, the researchers have used a correlation (Phi value of the connection) value between two different languages defined as:

$$\phi_{ij}=\frac{M_{ij}N-M_iM_j}{\sqrt{M_iM_j(N-M_i)(N-M_j)}}, \qquad (3)$$

where M_{ij} is the matrix which represents a number of translations, M_i - number of translations expressed in a particular language and N - total number of the translations in the dataset. The results created with that formula were limited by the relatively small sample size, as a consequence co-occurrence smaller than 6 were removed from the final network graph.

In our study, we included the entire network of the MIT's research (268 nodes, 545 edges). Each edge consisted of the set of attributes:

- "SourceLanguageCode" (ISO code of the translation's source language),
- "TargetLanguageCode"(ISO code of the translation's destination language),
- "SourceLanguageName"(name of the translation's source language),
- "TargetLanguageName"(name of the translation's destination language),
- "SourcePopulation"(population of the translation's source language),
- "TargetPopulation"(population of the translation's target language),
- "co-occurrences (number of co-occurrences between two languages) ",
- "Phi-correlation" (Phi value of the connection) ,
- "Tstatitisics" (statistical significance of the connection).

From the pool of information the dataset offered, we decided to select the ISO code of countries as nodes labels, edges were recreated using the source and target ISO codes. A summary of the network statistics can be seen in table 1; it gave a very dense network with a small diameter (2) and low average path length (1.05).

TABLE I. Book translations language network summary

Nodes	268
Edges	545
Average node degree (min, max)	4.07 (1, 110)
Network diameter	2
Average path length	1.05

The next step of our study was deeper analysis of the dataset; to achieve this goal we have used program called "Gephi" [9]. Our data analyses began with the degree distribution of our network. In the case of directed graph, this includes in- and out-degree as well as the combined degree of each node (Fig. 1).

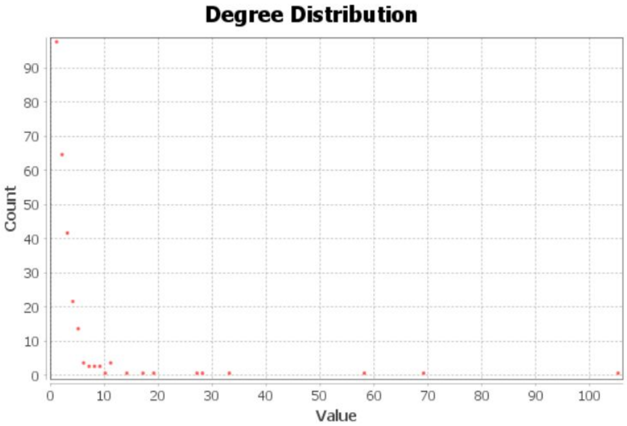

Fig. 1. The degree distribution (total) of nodes in a global book translations language network.

The graph showed the clear discrepancy between the node with the biggest degree (English) and the other languages. While the average degree for the whole dataset was 4.09, the value for English was 110. Two other nodes had quite large degrees (French, 69, and Russian, 58). This difference foreshadows the structure of the network - it will be English-centred (i.e. have a major hub) with few other sub-hubs, and many nodes being spokes in the network with few connections. Indeed, english had the largest Eigenvector centrality measure of 0.90 and French, the second largest (0.30).

There was also a very wide range of co-occurrence frequencies (i.e. edge weights) observed in the dataset, from a minimum of 6 to a maximum of 183329 attributed to the english-to-german translation edge. The mean co-occurrence frequency in the dataset was 2975.67.

For each language node, we additionally collected information about the language family, giving an additional node attribute. These data were retrieved manually from Wikipedia language pages (e.g. the *Japanese language* or *Azerbaijani language* pages), as also implemented by [7]. We took the highest group in the language family hierarchy, giving 29 unique families. There was an imbalance in the families present in the dataset; the largest families were Indo-European (38.5% of nodes, *n*=95), Niger-Congo (9.7%, *n*=24) and Turkic (8.9%, *n*=22). Whereas seven of the families occurred only once in the dataset (Yuman, Yukaghir, Siouan, Kartvelian, Hmong-Mien, Chukotko-Kamchatkan, and Arnhem). We also recorded four instances of *language isolate*, meaning that these languages have no identified genealogical relationships to other languages, i.e. there is no language family. The collected family data for each individual language can be found in appendix I.

IV. Approach

The chosen algorithms were implemented in Python using the iGraph package with preparation for the testing being fairly simple. Firstly, as mentioned in section III, the dataset comes with nine attributes, however, required for iGraph are only SourceNode, DestinationNode, and Weight. Therefore the unnecessary attributes were removed for efficiency and redundancy then loaded into the workspace using an iGraph function: Read_Ncol.

Community_walktrap in iGraph takes two optional parameters; weights and steps, where weights is a list containing edge weights and steps in the length of random walks to perform. The weights are automatically assigned as the third column with Read_Ncol, so to specify weights in the function, it is as simple as weights=g.es["weight"] where ‘g’ is the graph that was read in. Generally speaking, 3, 4, or 5 steps are used in implementation [5]. For this project, 4 steps were used.

Running community_walktrap returns a VertexDendrogram object which is initially cut at the maximum modularity. Running as_clustering() on this object returns a VertexClustering object, which describes the clustering of the vertex set of a graph. We are then able to plot this result in iGraph using plot(), with two parameters, mark_groups, and **visual_style, where mark_groups specifies whether to highlight some of the vertex groups by coloured polygons and visual_style specifying visual properties. We can simply set up a python dictionary containing keyword arguments that we wish to pass to plot(), which in this case, was marking the vertex labels. For the walktrap approach, the algorithm was first run whilst specifying weights, and then again without.

Label propagation is a simpler approach requiring no parameters, however, there are a few optional arguments; weights, initial, and fixed, where weights is identical to the structure in the walktrap approach. However, initial can be a list of the initial vertex labels, and fixed is a list of Booleans for each vertex with True corresponding to vertices who’s labeling should not change during the algorithm [7]. These last two parameters were not required with our testing, with only weights being specified in the arguments.

Unlike walktrap, label propagation does not return a dendrogram object and instead directly returns a VertexClustering object which can then be plotted in an identical fashion to the walktrap approach. Again, this algorithm was first run whilst specifying weights, and then again without.

V. Results

A. *Walktrap*

Initially, running the Walktrap algorithm weighted by the occurrence frequency identified three communities (fig. 2.A). There were two major communities, containing 71 and 94 nodes respectively, and a third, smaller community with only 3 members. The remaining 100 nodes (37.3%) were each assigned to their own cluster (i.e. *n*=1), which we did not count in our definition of a community.

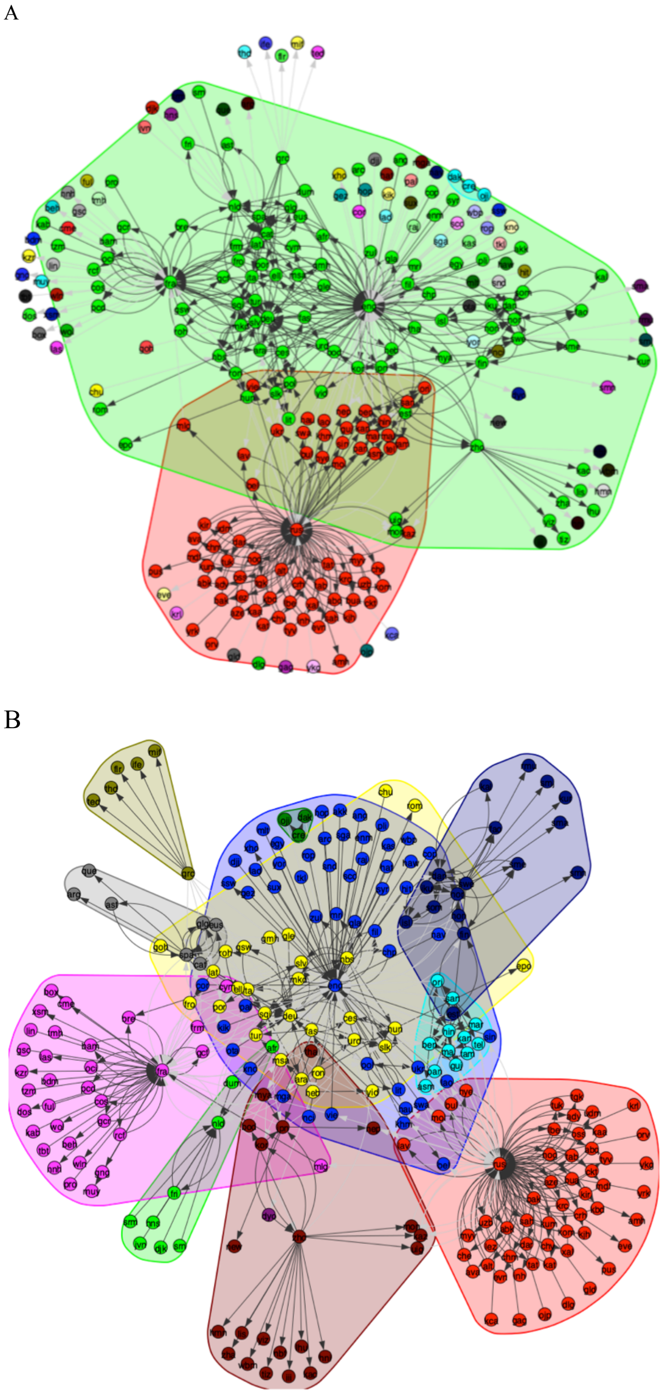

Fig. 2. The communities detected in the book translations global language network via Walktrap, with (A) and without (B) weighting the edges with the occurrence frequencies.

Repeating the test, this time giving all edges equal weighting yielded very different results (fig. 2.B). Ten communities were detected, with only a single node being assigned to its own cluster. There was also less of an imbalance in the size of the identified communities (mean 22.4 node members; minimum 3; maximum 54).

B. *Label propagation*

As found with the Walktrap algorithm, running label propagation on the weighted dataset identified two major communities of a similar size, with 70 and 102 members (fig. 3.A). There were, however, a further two minor communities, which contained 3 and 8 members respectively. Again, approximately one third of the nodes were not classified into communities due to the agglomerative nature of the approach (*n*=85, 31.7% of the dataset).

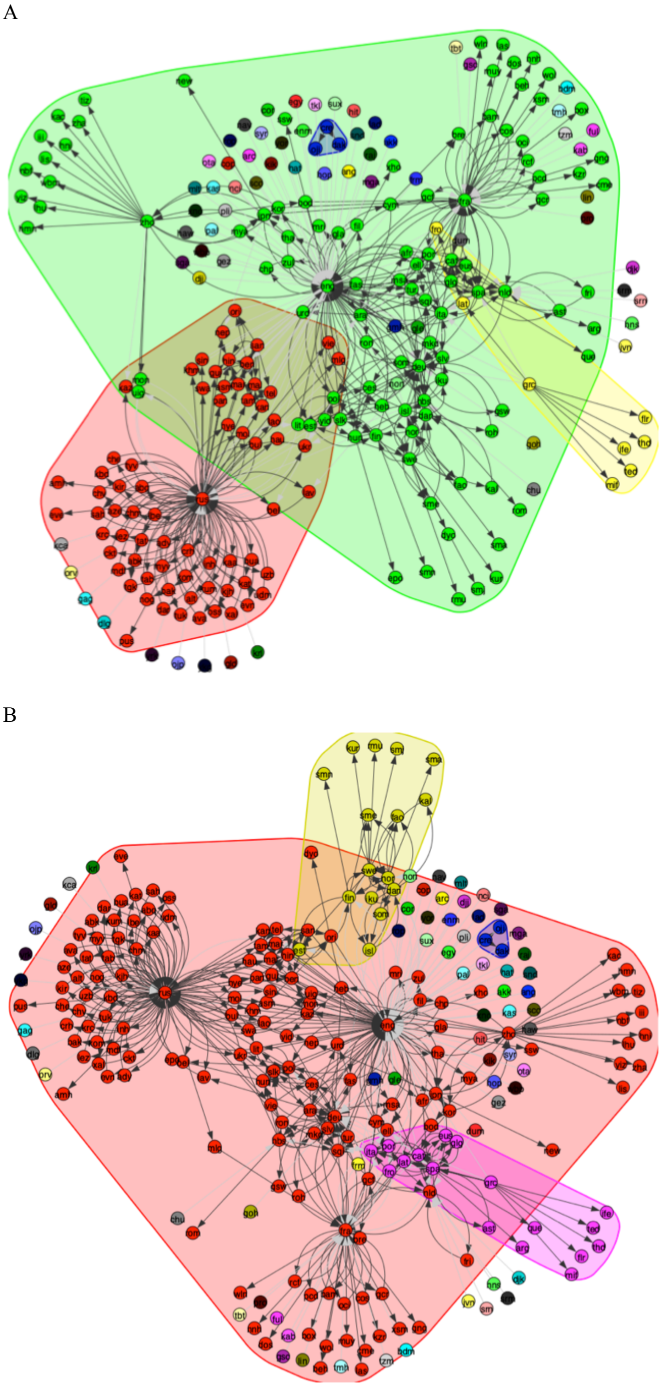

Fig. 3. The communities detected in the book translations global language network via label propagation, with (A) and without (B) weighting the edges with the occurrence frequencies. The graphs presented are an aggregated community structure of 50 runs.

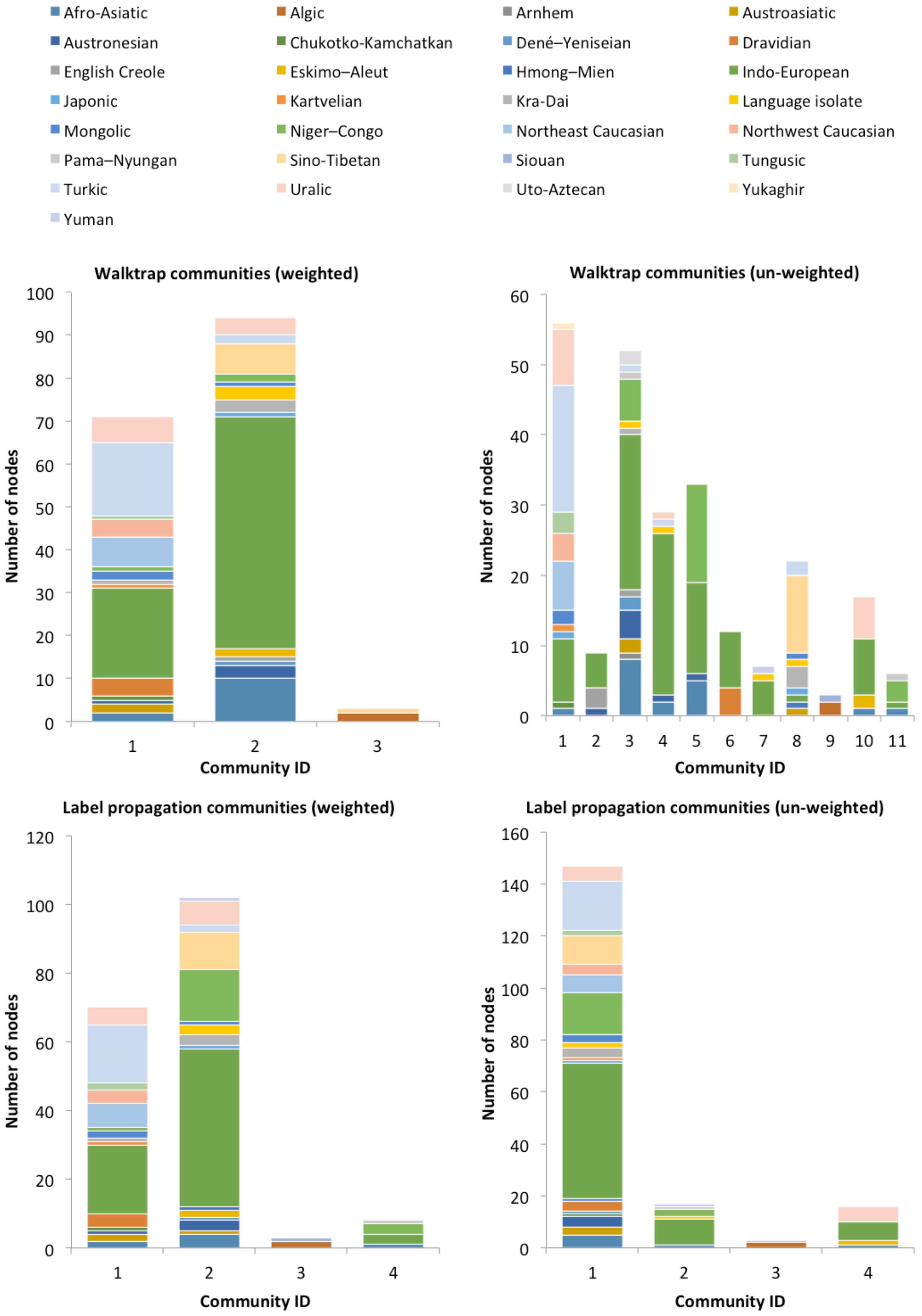

Fig. 3. The language families in the communities detected in the book translations global languages network using two algorithms: Walktrap and label propagation, with and without weighted edges by the co-occurrence frequencies.

Removing the edge weights and re-running the test was not as a significant difference as observed with the Walktrap approach. 85 of the nodes remained outside of the communities and, as when using weighting, four communities were still detected (fig. 3.B). However, one community was much larger than the others (n=147 in comparison to n=17, n=3 and n=16).

It was noticed that these results varied noticeably each time that the test was run. The label propagation process includes some random aspects, such as the order in which labels are assigned during each iteration or conflicts are handled (i.e. if two neighbouring labels have the the same, highest frequency then a one of the two is randomly chosen). This instability was not noticed when using the weighted edges - weighting adds stability to the process, making the results more reproducible. As such, we have visualised an aggregated community structure in fig. 3.

The high weights and dense nature of the graph has a considerable impact on how the algorithms work in this analysis. For example, for walktrap, in an unweighted graph, a node with 3 unweighted edges to neighbors would have a 1/3 chance to traverse each edge in a random walk, which is why we see many different communities in the unweighted version of the graph. However, when taking weights into account, the probability of choosing a highly weighted edge (mainly English and Russian) becomes extremely high, causing the walks to get stuck going back and forth on a single edge.

The situation with label propagation differs slightly; as we recall from section II, after each iteration, node labels update to that of the one shared by most of its neighbours. As Russian and English have such high in and out-degrees, the majority of nodes have one of these two languages in their neighborhood set, meaning there is a large possibility that these nodes will update labels to match that of the two aforementioned, even more so when edge weights are considered. This explains why we again see such large communities in both tests (weighted and unweighted).

C. Overlap with language families

For each of the communities detected, we also considered the overlap with language families. The results are presented in fig. 3. Upon visual inspection, we identified some common characteristics between the different detection methods compared. This did not seem to be influenced by the use of a weighted or unweighted network.

For example, in three of the tested methods, all Sino-Tibetan (peach, fig. 3) nodes belonged to the same community (Walktrap weighted community 2, LPA weighted community 2, and LPA unweighted community 1). In the Walktrap unweighted method, all but one Sino-Tibetan nodes were co-clustered.

The majority of nodes belonging to the Turkic language family (light blue) were also usually clustered within the same community - community 1 in all four of the tests in fig. 3. This same trend was also observed for the Northeast Caucasian languages. Given that both of these language families are spoken in similar parts of the world - Eastern Europe to Asia, the fact the they are always grouped in the same community suggests other factors to the language family, such as geographic location, may influence the GLCN.

The fact that the language family may not be the driving factor in GLCN community formation is further evidenced by some families that do not seem to be correlated with the node community identity at all. For example, nodes belonging to the Indo-European group (dark green) tend to be found in all of the communities detected.

VI. Conclusions

In this work, we have compared two agglomerative community detection algorithms using the book translations global languages network. We found that, when using the weighted network, neither approach generated many distinct communities in the dataset. Instead, generally, two major communities formed. This suggests that there are not distinct groups of languages, and instead, information sharing occurs at an international level in this network.

This conclusion is supported by the fact that we identified similarities in the two major communities identified by the two approaches, in terms of their size and constituents, by qualitatively considering the language families of the community members. Suggesting that the findings were not an artefact of the method employed.

By removing the edge weights, we found that this attribute of the underlying dataset was very important important in community formation. However, it is not known if the edge weights used in this work are truly representative of language co-occurrence frequencies. The edge weights were derived from data reported to UNESCO by many different international libraries [1]. These data may be subject to reporting biases of the various locations, with different locations following different practices and having differing levels of completeness, and may they may also not be up to date. This warrants further investigation.

In this work, we did not observe a link between the community identity and the language family of the nodes when visually inspecting the graphical data as has been statistically identified in other work [7]. As discussed in [7], other factors such as religion may be associated with language communities. In section V of this report, we hypothesised that this may be the case in this network. In future work, the correlation of the communities with other socio-political factors should be considered, to help understand the real-world influences and implications of potential communities in GLCNs.

An interesting extension would perhaps be to re-run the algorithms on the graph but without English and Russian nodes. We would expect that this would return a more interesting set of communities, that aren't quite as affected by the large bias towards highly connected nodes and heavily weighted edges.

This work was also limited by the dataset - book translations provide only a course representation of a true GLCN. The translation of a book is driven by publishing demand, which reflects the interests of literate populations only. However, it has been shown that the dataset shares some structural features with a GLCN derived from multi-lingual “tweeters”, including a positive correlation for co-occurrence frequencies [1]. Repeating the studies presented here in GLCNs derived from other sources should be the focus of future work, in order to compare these findings to those for GLCNs that capture other global populations, validating the communities detected in GLCNs.

References

[1] S. Ronen, B. Goncalves, K. Z. Hu, A. Vespignani, S. Pinker, C. A. Hidalgo, “Links that speak: the global language network and its association with global frame”, Proc. Nat. Acad. Sci. vol. E 111, pp. 5616-5622, 2014.

[2] S. Fortunato, D. Hric, “Community detection in networks: A user guide”, Physics Reports, vol 65(9), pp.1-44, 2016.

[3] P. Cimiano, A. Hotho and S. Staab, “Comparing Conceptual, Divisive and Agglomerative Clustering for Learning Taxonomies from Text”, European Conference on Artificial Intelligence (EPAI), 2004.

[4] M. Steinbach, G. Karypis, V. Kumar , “A Comparison of Document Clustering Techniques”, in KDD Workshop on Text Mining, 2000.

[5] X. Hu, W. He, H. Li, J. Pan, “Role-based Label Propagation Algorithm for Community Detection”, 2016. [online]. arXic.org. Available: https://arxiv.org/abs/1601.06307 [Accessed 17 Mar. 2019].

[6] Igraph.org, “python-igraph manual”, 2019. [online]. Available: https://igraph.org/python/doc/igraph.Graph-class.html#community_label_propagation [Accessed 15 Mar. 2019].

[7] A. Somoilenko, F. Karimi, D Edler, J Kunegis, M. Strohmaier, “Linguistic neighbourhoods: explaining cultural borders on Wikipedia through multilingual co-editing activity”, EPJ Data Science vol 5(9).

[8] Iso639-3.sil.org. (2019). *ISO 639-3*. [online] Available at: https://iso639-3.sil.org/ [Accessed 24 Mar. 2019].

[9] Gephi.org. (2019). *Gephi - The Open Graph Viz Platform*. [online] Available at: https://gephi.org/ [Accessed 25 Mar. 2019].

Appendix I

TABLE II. Language family data collected from Wikipedia

code	***Language node***	***Language family***
abk	Abkhazian	Northwest Caucasian
abq	Abaza	Northwest Caucasian
ady	Adyghe	Northwest Caucasian
afr	Afrikaans	Indo-European
akk	Akkadian	Afro-Asiatic
alt	Southern Altai	Turkic
amh	Amharic	Afro-Asiatic
ang	Old English (ca. 450-1100)	Indo-European
ara	Arabic	Afro-Asiatic
arc	Official Aramaic (700-300 BCE)	Afro-Asiatic
arg	Aragonese	Indo-European
asm	Assamese	Indo-European
ast	Asturian	Indo-European
ava	Avaric	Northeast Caucasian
aze	Azerbaijani	Turkic
bak	Bashkir	Turkic
bam	Bambara	Niger–Congo
bdm	Buduma	Afro-Asiatic
beh	Biali	Niger–Congo
bel	Belarusian	Indo-European
ben	Bengali	Indo-European
bod	Tibetan	Sino-Tibetan
box	Buamu	Niger–Congo
bre	Breton	Indo-European
bua	Buriat	Mongolic
bul	Bulgarian	Indo-European
cat	Catalan	Indo-European
ces	Czech	Indo-European
che	Chechen	Northeast Caucasian
chm	Mari (Russia)	Uralic
chp	Chipewyan	Dené–Yeniseian
chu	Church Slavic	Indo-European
chv	Chuvash	Turkic
ckt	Chukot	Chukotko-Kamchatkan
cme	Cerma	Niger–Congo
cop	Coptic	Afro-Asiatic
cor	Cornish	Indo-European
cos	Corsican	Indo-European
cre	Cree	Algic
crh	Crimean Tatar	Turkic
cym	Welsh	Indo-European
dak	Dakota	Siouan
dan	Danish	Indo-European
dar	Dargwa	Northeast Caucasian
deu	German	Indo-European
djj	Djeebbana	Arnhem
djk	Eastern Maroon Creole	English Creole
dlg	Dolgan	Turkic
dos	DogosÌ©	Niger–Congo
dum	Middle Dutch (ca. 1050-1350)	Indo-European
dyo	Jola-Fonyi	Niger–Congo
egy	Egyptian (Ancient)	Afro-Asiatic
ell	Modern Greek (1453-)	Indo-European
eng	English	Indo-European
enm	Middle English (1100-1500)	Indo-European
epo	Esperanto	Language isolate
est	Estonian	Uralic
eus	Basque	Language isolate
eve	Even	Tungusic
evn	Evenki	Tungusic
fao	Faroese	Indo-European
fas	Persian	Indo-European
fil	Filipino (macrolanguage)	Austronesian
fin	Finnish	Uralic
flr	Fuliiru	Niger–Congo
fra	French	Indo-European
fri	Western Frisian	Indo-European
frm	Middle French (ca. 1400-1600)	Indo-European
fro	Old French (842-ca. 1400)	Indo-European
ful	Fulah	Niger–Congo
gag	Gagauz	Turkic
gcf	Guadeloupean Creole French	Indo-European
ger	Guianese Creole French	Indo-European
gez	Geez	Afro-Asiatic
gla	Scottish Gaelic	Indo-European
gld	Nanai	Tungusic
gle	Irish	Indo-European
glg	Galician	Indo-European
gmh	Middle High German (ca. 1050-1500)	Indo-European
gng	Ngangam	Niger–Congo
goh	Old High German (ca. 750-1050)	Indo-European
grc	Ancient Greek (to 1453)	Indo-European
gsc	Gascon	Indo-European
gsw	Swiss German	Indo-European
guj	Gujarati	Indo-European
hat	Haitian	Indo-European
hau	Hausa	Afro-Asiatic
haw	Hawaiian	Austronesian
hbs	Serbo-Croatian	Indo-European
heb	Hebrew	Afro-Asiatic
hin	Hindi	Indo-European
hit	Hittite	Indo-European
hmn	Hmong	Hmong–Mien
hni	Hani	Sino-Tibetan
hns	Caribbean Hindustani	Indo-European
hop	Hopi	Uto-Aztecan
hun	Hungarian	Uralic
hye	Armenian	Indo-European
ife	IfÌ¬	Niger–Congo
iii	Sichuan Yi	Sino-Tibetan
iku	Inuktitut	Eskimo–Aleut
inh	Ingush	Northeast Caucasian
isl	Icelandic	Indo-European
ita	Italian	Indo-European
jpn	Japanese	Japonic
jvn	Caribbean Javanese	Austronesian
kaa	Kara-Kalpak	Turkic
kab	Kabyle	Afro-Asiatic
kac	Kachin	Sino-Tibetan
kal	Kalaallisut	Eskimo–Aleut
kan	Kannada	Dravidian
kas	Kashmiri	Indo-European
kat	Georgian	Kartvelian
kaz	Kazakh	Turkic
kbd	Kabardian	Northwest Caucasian
kca	Khanty	Uralic
khm	Central Khmer	Austroasiatic
kik	Kikuyu	Niger–Congo
kir	Kirghiz	Turkic
kjh	Khakas	Turkic
kom	Komi	Uralic
kor	Korean	Language isolate
krc	Karachay-Balkar	Turkic
krl	Karelian	Uralic
kum	Kumyk	Turkic
kur	Kurdish	Indo-European
kzr	Karang	Niger–Congo
lad	Ladino	Indo-European
lao	Lao	Kra-Dai
las	Lama (Togo)	Niger–Congo
lat	Latin	Indo-European
lav	Latvian	Indo-European
lbe	Lak	Northeast Caucasian
lez	Lezghian	Northeast Caucasian

lhu	Lahu	Sino-Tibetan
lin	Lingala	Niger–Congo
lis	Lisu	Sino-Tibetan
lit	Lithuanian	Indo-European
mal	Malayalam	Dravidian
mar	Marathi	Indo-European
mdf	Moksha	Uralic
mga	Middle Irish (900-1200)	Indo-European
mif	Mofu-Gudur	Afro-Asiatic
mkd	Macedonian	Indo-European
mlg	Malagasy	Austronesian
mlt	Maltese	Afro-Asiatic
mol	Moldavian	Indo-European
mon	Mongolian	Mongolic
mri	Maori	Austronesian
msa	Malay (macrolanguage)	Austronesian
muy	Muyang	Afro-Asiatic
mya	Burmese	Sino-Tibetan
myv	Erzya	Uralic
nav	Navajo	Dené–Yeniseian
nbf	Naxi	Sino-Tibetan
nci	Classical Nahuatl	Uto-Aztecan
nep	Nepali macrolanguage	Indo-European
new	Newari	Sino-Tibetan
nld	Dutch	Indo-European
nnh	Ngiemboon	Niger–Congo
nog	Nogai	Turkic
non	Old Norse	Indo-European
nor	Norwegian	Indo-European
oci	Occitan (post 1500)	Indo-European
oji	Ojibwa	Algic
ojp	Old Japanese	Japonic
ori	Oriya (macrolanguage)	Indo-European
orv	Old Russian	Indo-European
oss	Ossetian	Indo-European
ota	Ottoman Turkish (1500-1928)	Turkic
pal	Pahlavi	Indo-European
pan	Panjabi	Indo-European
pcd	Picard	Indo-European
pli	Pali	Indo-European
pol	Polish	Indo-European
por	Portuguese	Indo-European
pro	Old ProvenÌ_al (to 1500)	Indo-European
pus	Pushto	Indo-European
que	Quechua	Yuman
raj	Rajasthani	Indo-European
rcf	RÌ©union Creole French	Indo-European
rmu	Tavringer Romani	Indo-European
roh	Romansh	Indo-European
rom	Romany	Indo-European
ron	Romanian	Indo-European
rop	Kriol	English Creole
rus	Russian	Indo-European
sah	Yakut	Turkic
san	Sanskrit	Indo-European
sco	Scots	Indo-European
sga	Old Irish (to 900)	Indo-European
sin	Sinhala	Indo-European
slk	Slovak	Indo-European
slv	Slovenian	Indo-European
sma	Southern Sami	Uralic
sme	Northern Sami	Uralic
smj	Lule Sami	Uralic
smn	Inari Sami	Uralic
snd	Sindhi	Indo-European
som	Somali	Afro-Asiatic
spa	Spanish	Indo-European
sqi	Albanian	Indo-European
srm	Saramaccan	English Creole
srn	Sranan Tongo	English Creole
ssw	Swati	Niger–Congo
sux	Sumerian	Language isolate
swa	Swahili (macrolanguage)	Niger–Congo
swe	Swedish	Indo-European
syr	Syriac	Afro-Asiatic
tab	Tabassaran	Northeast Caucasian
tam	Tamil	Dravidian
tat	Tatar	Turkic
tbt	Tembo (Kitembo)	Niger–Congo
ted	Tepo Krumen	Niger–Congo
tel	Telugu	Dravidian
tgk	Tajik	Indo-European
tha	Thai	Kra-Dai
thd	Thayore	Pama–Nyungan
tiz	Tai Hongjin	Kra-Dai
tkl	Tokelau	Austronesian
tmh	Tamashek	Afro-Asiatic
tuk	Turkmen	Turkic
tur	Turkish	Turkic
tyv	Tuvinian	Turkic
tzm	Central Atlas Tamazight	Afro-Asiatic
udm	Udmurt	Uralic
uig	Uighur	Turkic
ukr	Ukrainian	Indo-European
urd	Urdu	Indo-European
uzb	Uzbek	Turkic
vie	Vietnamese	Austroasiatic
wbm	Wa	Austroasiatic
wbp	Warlpiri	Pama–Nyungan
wln	Walloon	Indo-European
wol	Wolof	Niger–Congo
xal	Kalmyk	Mongolic
xho	Xhosa	Niger–Congo
xno	Anglo-Norman	Indo-European
xsm	Kasem	Niger–Congo
yid	Yiddish	Indo-European
yiz	Azhe	Sino-Tibetan
ykg	Northern Yukaghir	Yukaghir
yor	Yoruba	Niger–Congo
yrk	Nenets	Uralic
zha	Zhuang	Kra-Dai
zho	Chinese	Sino-Tibetan
zul	Zulu	Niger–Congo