Venue Matching in Social Network APIs using Neural Networks

Vassilios Kalavrouziotis Andreas Komninos John Garofalakis
Computer Engineering & Informatics Department
University of Patras
Greece
[kalabrouzi, akomninos, garofala]@ceid.upatras.gr

ABSTRACT

A multitude of social media APIs from popular services such as Facebook, Twitter and Google, allow programmers access to user generated data that is pertinent to physical venues represented within these services. In our paper, we attempt to address the issue of automatically matching venue representations from these diverse APIs, in order to obtain a more complete representation of user cyber-physical interaction with these venues. We present our work comparing a neural network approach against Nearest Point and Longest Common Substring algorithms.

CCS CONCEPTS

• **Information systems ~ Information extraction** • **Information systems ~ Clustering and classification** • Information systems ~ Web searching and information discovery • Information systems ~ Data extraction and integration • Information systems ~ Social networks

KEYWORDS

Social Networks, POI matching, Machine Learning

ACM Reference format:

Vassilios Kalavrouziotis, Andreas Komninos and John Garofalakis. 2018. Venue Matching in Social Network APIs using Neural Networks. In *Proceedings of ACM 22nd Pan-Hellenic Conference on Informatics (PCI'18). ACM, New York, NY, USA, 6 pages.* https://doi.org/10.1145/3291533.3291558.

PCI '18, November 29-December 1, 2018, Athens, Greece

ACM ISBN 978-1-4503-6610-6/18/11…$15.00
https://doi.org/10.1145/3291533.3291558

1 Introduction

Due to the utility and popularity of social networks (SNs), users generate large volumes of data daily, much of which is geo-tagged either using precise coordinates, or by co-tagging a spatially positioned venue that is already represented in the SN, leading to the concept of a Location Based SN (LBSN). In a sense, it can be said that users can interact with their physical environment using LBSNs, indicating their presence or appraisal of a venue in a spatiotemporal context [1, 2]. A common activity that includes such cyber-physical interaction is the "check-in", i.e. the explicit action of indicating current presence in a venue, the "like" or "rating" of a venue, as well as leaving a "tip" for other users, or "tagging" a photo, status or other update with the venue. For researchers, this data is valuable as it can be used to capture the urban dynamics of an area of interest or be used for other analyses.

As users are often registered with and employ various SNs, it is difficult to obtain a complete picture of their cyber-physical interactions, since venue representations amongst LBSN services are not in any way linked. A further problem is that because venue representations are often generated by the users themselves and are not moderated, there is often great discrepancy between the representation of a venue amongst diverse LBSNs. For example, the same venue might be represented with different coordinates with a varying degree of discrepancy (slight inaccuracies in the location sensor of the device used to create the venue profile, or large inaccuracies because a venue has moved to new premises and this change is not reflected across all networks). The same applies to venue names, which may vary because of different spellings (e.g. "Aróe" and "Aroe"), omissions of common words (e.g. "Tag Café" and "Tag") or the re-naming of a business as it changes ownership over time. Venues that are closed are also a source of clutter. Finally, issues arise due to the multiple representations of the same venue in a LBSN. To address some of these issues, researchers have proposed systems that use humans as an information source. For example, in [3], a system is proposed that allows social network users to ask other users questions about venues, leveraging from the expertise or knowledge of others. However, their analysis shows that there are issues that

relate to response time for the questions posed, as approximately 75% of questions were responded after 10 minutes. Additionally, approximately 65% of questions received up to 3 responses only and there was no way for the user to rate the response received (some responses were also completely useless, e.g. "I don't know"). This impacts on the trustworthiness of such an approach. Another, more automated approach that combines data gathered automatically from social networks (Twitter and Foursquare) and considers the "expertise" of individual users, based on their geo-located tweets and venue check-ins, is presented in [4]. Venues for a particular goal (e.g. find a good seafood restaurant) are thus presented as suggestions to the user after matching the query with relevant venues. The researchers' analysis showed that compared to user-review based, expert-based and hybrid schemes, this approach offers very good results without the cost of paying human reviewers. However, a deficit of this approach is that it effectively augments the current Foursquare system, adding a layer of tweets and tags pertinent to the venues only represented therein. Hence it doesn't really address the issue of matching venues represented in multiple SNs. Other approaches, aiming to offer better venue suggestions by analyzing not just the check-in statistics but also user profiles, are offered by [5], who augment the Gowalla dataset with calculated additional metrics to improve suggestions. A good survey of venue recommendation methods is provided by Rosi et al. [6], in which it is apparent that techniques for automatically matching venues from multiple SN datasets are not present in literature. To addess this shortcoming, Celino et al. [7] proposed a solution to matching venues using a crowdsourcing approach. They presented a pervasive game called UrbanMatch, in which users are called upon to provide links between POIs represented in Flickr and OpenStreetMap. Clearly though this approach requires a number of active players, who, without incentive, could not possibly cover the entire globe.

We set out to examine whether it might be possible to automate the process of matching venue representations across diverse LBSNs, in order address a limitation present in all current research in LBSN use, which is focused on mining data from single networks [8][9]. So far, research into this issue has been very limited. Mashhadi et al. [10] attempted a pairing of POIs present in OpenStreetMap and NavTeq datasets, using the criteria of geographic distance and Levenshtein distance of the POI names. They empirically derived a threshold of 100 meters and 0.33 for the normalised Levenshtein distance as a criterion with which to match POIs. They concluded that this scheme has a 97% accuracy but their tests were performed on a very small dataset of 30 POIs. Scheffler et al. [11] have used a geographical distance filter followed by string processing (Levenshtein distance) on the venue names, using a threshold of 10% to match venues. Comparing this approach with a Nearest Point (NP) and a Longest Common Substring (LCS) approach, a distinct advantage was found in their approach, which achieved up to 79% accuracy, however this was on a limited subset of 50 random POIs from Facebook and Qype, of which 34 and 33 respectively were used as training data. McKenzie et al. [12] also attempted to match Foursquare and Yelp POIs using the venue name Levenshtein distance, phonetic similarity, category matching and geographical location in a weighted multi-attribute model. They obtained 97% accuracy in matching using 200 POIs from both services, in a dataset of verified matching POIs. This performance is drastically reduced when the datasets contain non-matching POIs, returning a high percentage of true matches (up to 95%) but also many false matches (up to 65%).

2 Using Neural Networks to Match Venues

Since the approaches previously used in literature did not employ a machine learning approach, we decided to implement a neural network-based approach and evaluate its performance against the NP and LCS approaches, which can be considered as the baseline measure for performance. We used the Tremani Neural Network framework[1], written in PHP, in order to develop a solution that would be easy to run on a server environment and integrate with various web services. The implemented network is a feed forward multi-level network, using up to two intermediate (hidden) levels.

2.1 Training and Test Data

We collected a set of data from Foursquare and Facebook, as described in [13], for the city of Patras, Greece. We found 403 distinct venues from Facebook and 1777 venues from Foursquare in the area of interest. We proceeded to manually match 240 pairs of venues from both datasets and also create a further 240 pairs of false matches. For each of the pairs, we calculated their geographic distance in meters, using the Halversine distance function, and also the Levenshtein distance of the venue names, converting Greek character in venue names to Latin, where necessary. A sample of this training set is shown in Tables 1 and 2 below.

Table 1: Sample matching venues from training set

Foursquare name	*Facebook name*	*Lev. Dist*	*Geo. Dist (m)*
Queen	@queen psila alwnia	13	14
Avantaz	Abantaz Club Ellinadiko	17	48
Abbey Kitchen Bar	Abbey Cafe	10	22
Amelie	Amelie vintage cafe	13	11
b.b.king	Bb King	1	33
bibliotheca | sala di studio	Bibliotheca	16	8

Table 1 shows some characteristic examples of the issues in venue matching from different social networks. On the first row, Foursquare displays the "common" name for a venue, while on Facebook, it is listed as the owner intended ("@queen"), followed

[1] Tremani Neural Network: http://neuralnetwork.sourceforge.net/ [accessed 20/6/2014]

by the name of the plaza this venue is located in ("psila alwnia"). The plaza name itself is a "Greeklish" (Greek written with English, i.e. Latin characters) rendition of the proper name ("ψηλά αλώνια"), where the omega ("ώ") character is represented by an English lookalike letter ("w").

Table 2: Sample non-matching venues from training set

Foursquare name	*Facebook name*	*Lev. Dist*	*Geo. Dist (m)*
pantelospito	Plan B Club	10	1159
Dasullio - The Bright Site Of City	Moulin Rouge	27	1391
Souvlakeri	Bardot	9	613
RF street	Onisimon	8	124
X-Treme Stores	Exte Hair Design	12	895
Vodafone	Diamond Event Planners	18	256

To form an unbiased training set, we randomly chose two thirds of each pair category (true and false matches), resulting in a training set of 320 pairs. We fed the training data into a network, consisting of 3 input neurons (spatial distance, Levenshtein distance and a polarized variable with an initial value of 1), 2 hidden neurons and 1 output neuron. Because we observed that venue characteristics with large attribute values affect the feedback process greatly, we normalized the input data with a linear scaling transformation [14]. In our transform, the formula used is

$$I = I_{min} + (I_{max} - I_{min}) * (D - D_{min}) / (D_{max} - D_{min}),$$

where I is the value after normalization, I_{min} and I_{max} represent the normalization range (in our case -1.0 and 1.0 respectively), D is the value before normalization and D_{min} and D_{max} are the minimum and maximum values in our sample data.

2.2 Performance of the Neural Network

To test the neural network's performance, we ran it on the remaining 160 pairs from our original set, for which we know the correct classification. In order to configure the NN's learning behaviour on the task, we experimented with different learning rate and momentum combinations. In [15], it is suggested that learning rates are typically set to 0.1 and momentum values to 0.9. We explored the region around these settings, trying out all the possible combinations for learning rate L∈[0.05, 0.25] and momentum M∈[0.75, 0.95], in steps of 0.05 for each value. This resulted in 25 L-M value combinations. For each combination we trained the network 20 iterations and recorded the accuracy of the NN in correctly classifying whether a venue pair is matching or non-matching. For each iteration of training, we allowed a maximum of 3 attempts (rounds), which meant that if training was not successful (i.e. it met our maximum squared error threshold of 0.5), it could be attempted for a total of up to 3 times.

From this analysis, it is obvious that as both learning rate value grows, momentum seems to have very little impact on the accuracy of the predictions, up till the combination L,M=[0.15, 0.9]. From this point onwards, as the learning rate increases, the combinations with any momentum value bring instability to the NN, as indicated not only by the lower accuracy scores, but also by their larger standard deviations. When taking into consideration the number of training rounds required to successfully complete training, it is clear that the operational parameters of the network offer increased accuracy and less time to successfully train for L∈[0.05, 0.1] and momentum M∈[0.75, 0.95]. From these combinations, we chose to continue with L,M=[0.1, 0.8], as this combination offers a good compromise between accuracy and trainability.

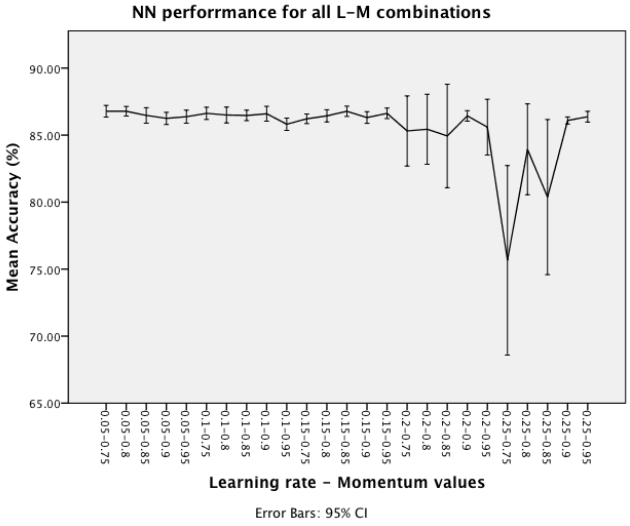

Figure 1. Performance of NN for all L-M combinations

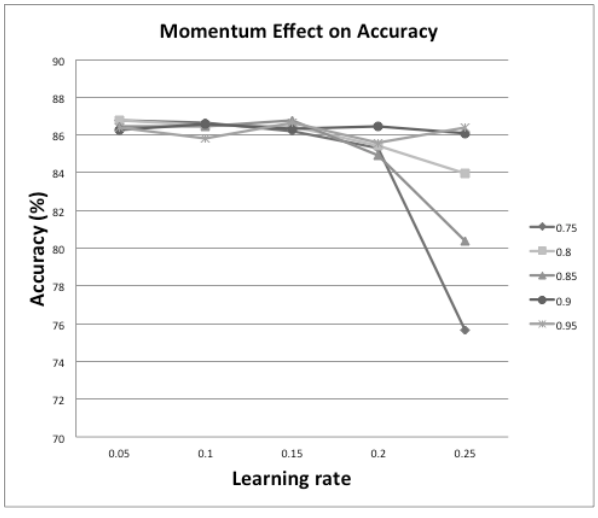

Figure 2. Effect of momentum values on the accuracy of the NN

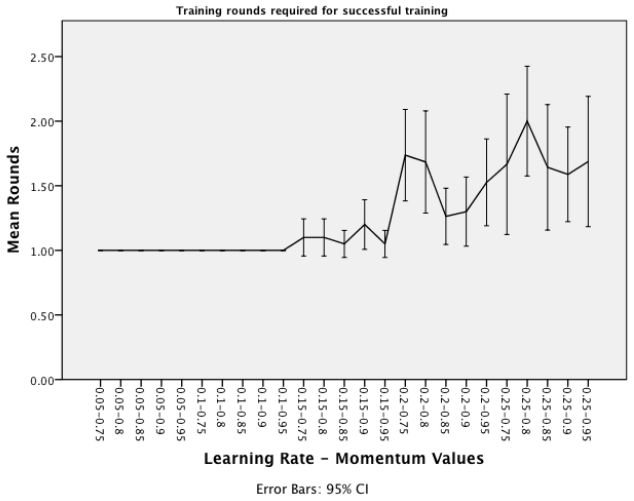

Figure 3. Training rounds required to successfully train the NN for all L-M combinations

Finally, we experimented with the number of hidden layers and the neurons within them. First, using one hidden layer, we experimented with the number of neurons, running a training and accuracy test 15 times for each value of neurons ranging from 1 to 10 (

Figure 4). An ANOVA of the results indicated that there was no statistically significant difference in the accuracy obtained (p=0.725). The best accuracy was obtained with two hidden nodes (M=86.79, SD=0.88).

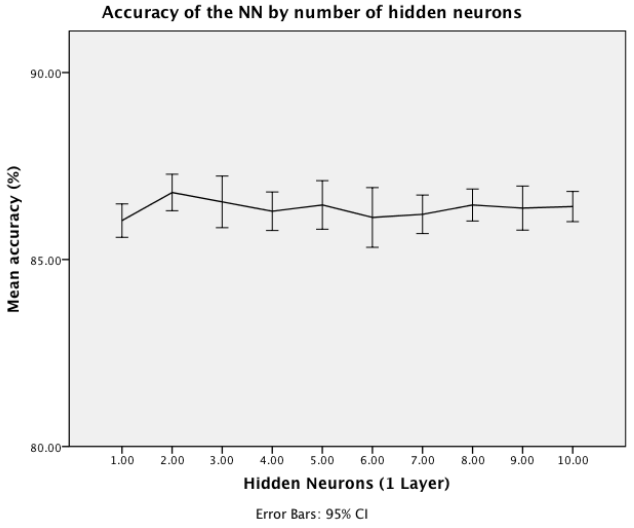

Figure 4. Effect of number of hidden neurons in a single layer on our NN accuracy

Subsequently we experimented with the number of hidden layers (H) and neurons (N) and examined the combinations HxN of [1x6], [2x3], [3x2] and [6x1]. An ANOVA confirmed that the difference in performance for all these combinations is statistically significant (p<0.05), clearly setting apart the configurations of [1x2] (from our previous experiment) and [1x6], [2x3] from the rest of the configurations (Figure 5). A further ANOVA between these three configurations did not show a statistically significant result (p=0.757), hence we proceeded with configuring our NN with a learning rate of 0.1, momentum of 0.8, one hidden layer with two neurons.

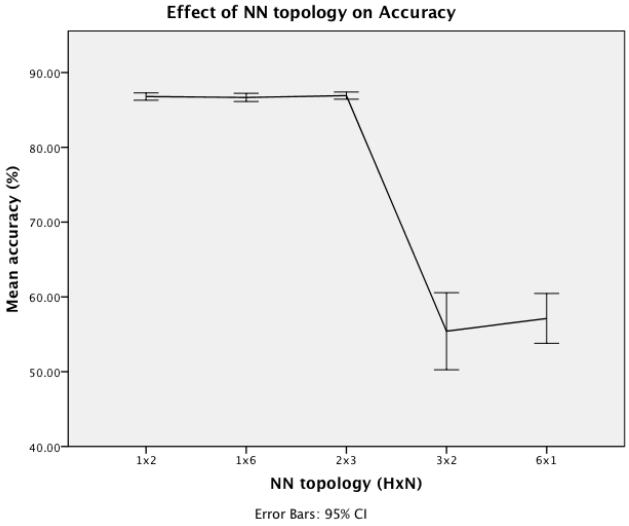

Figure 5. Effect of NN topology hidden layers and neurons (HxN) on our NN accuracy

With these settings, we take a closer look into how the NN performed with our data. Using the 320 Facebook venues, we attempted to match these to a further set of 320 random venues from Foursquare. A manual inspection showed that 167 of the Facebook venues did actually match a venue in the Foursquare set. To evaluate the NN performance, we paired each Facebook venue with each Foursquare venue, ran the pair into the NN and for each Facebook venue, we recorded the paired Foursquare venue for which the NN provided the best score (i.e. confidence on whether it matched or not), ranging from [1, -1]. From these tests, we found that the NN was able to correctly match 133 of the 167 truly matching venues (79.6%). For a further 34 venues (20.4%), while a match did exist, the NN was not able to find it.

For the correctly classified pairs, we note that the average score provided for the best matching pair was 0.55 but there is quite a wide fluctuation in the results (SD=0.52). Nevertheless we note that 69.2% of the correctly matched venues are obtained with a score of 0.5 or greater (Figure 6). Using this score figure as a threshold, the NN correctly identified a match for 92 of the Facebook venues (55%) but also incorrectly identified a match for 13 Facebook venues (7.8%).

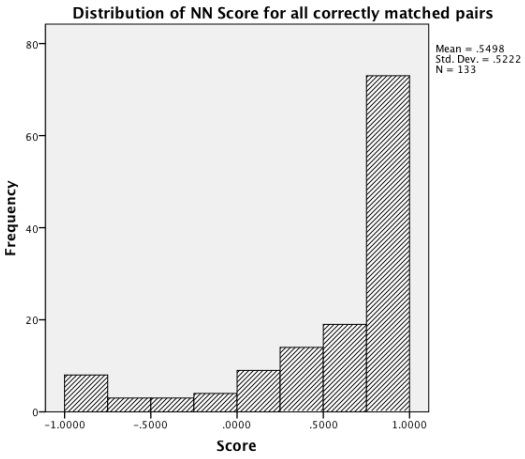

Figure 6. Distribution of NN score for all correctly matched pairs

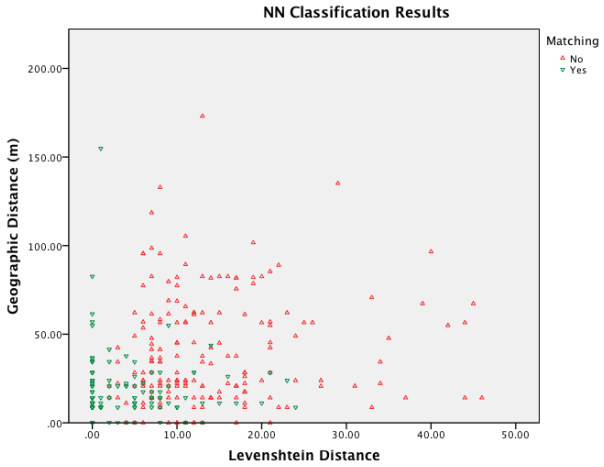

Figure 7: Classification results against Geographic and Levenshtein distance values

The classification results for the entire test set as a function of geographical distance and Levenshtein distance are shown in Figure 7. The results for those pairs reported with a score of at least 0.5 is shown in Figure 8, where we note that problems seem to arise because of the increase of Levenshtein distance in the pairs. The 92 correctly identified pairs exhibit a mean geographical distance of 16.03m (SD=12.14m) and Levenshtein distance of 1.02 (SD=1.74). For the 13 wrongly identified pairs, the respective values are 15.37m (SD=12.86m) and 4.0 (SD=1.47). A Mann-Whitney test reveals a statistically significant difference ($p<0.01$) only for the Levenshtein distance, revealing that the issue here is the venue names, which happen to be close geographically but also quite close as far as Levenshtein distances are concerned. As a comparison measure, when considering all those pairs where the best match is reported with a confidence of <0.5, the mean Levenshtein distance for non-matching pairs (173) is 14.78 (SD=8.90). For this said set of venues, where a match was successfully found (41 cases), a statistically significant smaller Levenshtein distance was found at a value of 10.05 (SD=5.32) ($p<0.01$).

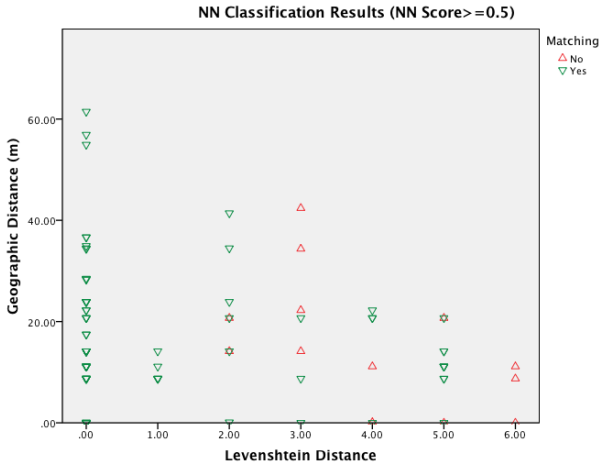

Figure 8: Classification results against Geographic and Levenshtein distance values (score>=0.5)

Finally, we turn our attention to those 34 instances where the NN failed to find an appropriate match for the Facebook venues but such a match did, in fact, exist in the Foursquare set. In these instances we observe that the mean geographical distance between the points is very large (116.53m, SD=150.11m) and that the mean Levenshtein distance is very close to that of pairs where an appropriate match was not found (11.21, SD=7.90). It is clear for these results that the determining factor, which caused these possible matches to be discarded (i.e. to yield a lower score in the NN), was the large distance of the venues in the different datasets. The larger Levenshtein distance can be attributed to the inclusion in several cases of descriptive words (e.g. "Stone Bar" vs. "Stone") or alternative word order (e.g. "Happy Café" vs. "Café Happy").

2.2 Comparative evaluation with NP and LCS

As a last step, we proceeded to examine our NN approach against the baseline cases of Nearest Point (NP) and Longest Common Substring (LCS). We used a set of 480 venues, of which 240 were true matches and 240 pairs false matches. The NP algorithm was successful in identifying 47.92% of the cases correctly, while this percentage rose to 78.33% for LCS. The NN approach obtained an accuracy score of 83.75%, clearly performing better than the other two. We further proceeded to include the remaining of the 1777 Foursquare venues into the trial, matching them randomly with Facebook venues, thus significantly increasing the number of false matches. We found that the accuracy in classifying the pairs as matching or non-matching fell significantly to 30.8% for NP, remained practically the same with 78.3% for LCS and declined also slightly for NN (83.3%). The difference in accuracy between NP and LCS was statistically significant ($p<0.01$), while the same cannot be said for the difference between LCS and NN, though the p value was

close to statistical significance (p=0.06). These results again highlight the importance of venue naming as a measure for determining matching venues, as it is clear that geographical distance alone is not sufficient as a measure and LCS outperforms NP.

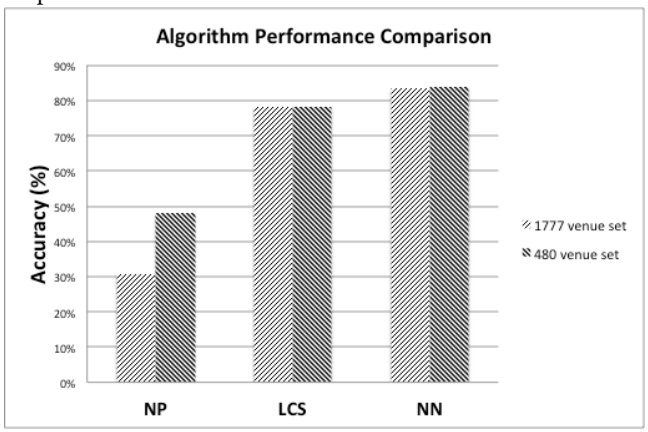

Figure 9. Performance comparison vs. NP and LCS

3 Discussion and Future Work

As can be seen from our results, neural networks show significant promise in solving the problem of matching POIs from distinct social networks with considerable success. Our approach yields a better result than Scheffler et al. [11] and, at a threshold of 0.5, largely avoids the problem of returning false matches as encountered by McKenzie et al. [12]. However as an approach, it does have some limitations, which we will now discuss.

Firstly, we used a dataset which contained many Greek names. To address the issue of several venues being named in Latin script in one SN but Greek in the other, we transformed all names using a "Greeklish" function, which is imperfect, as there is no "standard" for writing "Greeklish" and hence our ruleset can produce different results from the way an individual might spell a venue name in "Greeklish". This introduces uncertaintly in our NN which a human user might not face. Unfortunately, because we needed to manually match our data based on our local experience, we could not avoid this issue.

Continuing, our NN was trained on data from one city only, hence we cannot generalize our results for other cities. This is because SN use in different cities can vary (some cities may not have as many venues represented, or may have many more that are closer packed together). Further, our approach uses the Levenshtein distance, which, in itself, is a simple measure for comparing strings.

We are currently investigating other techniques which may yield additional input for the NN, such as phonetic similarity. We are also considering a transformation filter which will remove some common keywords, such as venue type descriptors (e.g. "café", "restaurant") from venue names, as well as duplicate words in the venue name. Our NN might be improved in the future by adding further information values into the neural network input neurons. As an example, we might use the venue category descriptors (although matching the categories represented in varying SNs is another issue), or the number of likes, check-ins and tags that this venue has. The latter seems like a promising element, as in our dataset, we have found that the total number of Facebook "likes" and Foursquare "check-ins" for matching venues is strongly correlated (Spearman's R=0.208, p<0.01). We are also exploring linkages between the number of "tags" and unique users tagged in a venue.

As a last step, we are considering the development of a mobile application that will leverage from this functionality, in an architecture that offloads computation and matching to a central server. We envisage users being better informed in a locality about the social semantics of venues, by offering a unified view of SN statistics for a particular place, hence enhancing their trust in the information provided. Our desire is to also implement an explicit feedback mechanism which will allow users to manually suggest or correct matches to venues, hence providing further feedback and training data to our NN-driven system.

REFERENCES

[1] Cho, E., Myers, S. A. and Leskovec, J. 2011. Friendship and mobility: user movement in location-based social networks. In Proc. 17th ACM SIGKDD international conference on Knowledge discovery and data mining ACM (2011), 1082-1090.

[2] Ferrari, L., Rosi, A., Mamei, M. andZambonelli, F. 2011. Extracting urban patterns from location-based social networks. In *Proc. 3rd ACM SIGSPATIAL International Workshop on Location-Based Social Networks*, ACM (2011), 9-16.

[3] Liu, Y., Alexandrova, T. and Nakajima, T. 2013. Using stranger as sensors: temporal and geo-sensitive question answering via social media. In Proceedings of the 22nd international conference on World Wide Web (pp. 803-814). International World Wide Web Conferences Steering Committee.

[4] Shankar, P., Huang, Y. W., Castro, P., Nath, B. and Iftode, L. 2012. Crowds replace experts: Building better location-based services using mobile social network interactions. In Pervasive Computing and Communications (PerCom), 2012 IEEE International Conference on (pp. 20-29). IEEE.

[5] Ying, J. J. C., Lu, E. H. C., Kuo, W. N. and Tseng, V. S. 2012. Urban point-of-interest recommendation by mining user check-in behaviors. In Proceedings of the ACM SIGKDD International Workshop on Urban Computing(pp. 63-70). ACM.

[6] Rosi, A., Mamei, M. and Zambonelli, F. 2013. Integrating social sensors and pervasive services: approaches and perspectives. International Journal of Pervasive Computing and Communications, 9(4), 294-310.

[7] Celino, I., Contessa, S., Corubolo, M., Dell'Aglio, D., Della Valle, E., Fumeo, S. and Krüger, T. 2012. UrbanMatch-linking and improving Smart Cities Data. In LDOW.

[8] Zhang, A. X., Noulas, A., Scellato, S. and Mascolo, C. 2013. Hoodsquare: Modeling and Recommending Neighborhoods in Location-based Social Networks. In Proc. IEEE Conference on Social Computing (SocialCom), IEEE (2013), 69-74).

[9] Li, Y., Steiner, M., Wang, L., Zhang, Z. L. and Bao, J. 2013. Exploring venue popularity in foursquare. In Proc. IEEE INFOCOM 2013, IEEE (2013), 3357-3362

[10] Mashhadi, A., Quattrone, G. and Capra, L. 2013. Putting ubiquitous crowd-sourcing into context. In Proceedings of the 2013 conference on Computer supported cooperative work (pp. 611-622). ACM.

[11] Scheffler, T., Schirru, R., and Lehmann, P. 2012. Matching points of interest from different social networking sites. In KI 2012: Advances in Artificial Intelligence, Springer Berlin Heidelberg (2013), 245-248.

[12] McKenzie, G., Janowicz, K. and Adams, B. 2014. A weighted multi-attribute method for matching user-generated Points of Interest. Cartography and Geographic Information Science, Taylor & Francis (2014), 1-13 (ahead-of-print)

[13] Komninos, A., Besharat, J., Stefanis, V., & Plessas, A. 2013. Capturing Urban Dynamics with Scarce Check-In Data. IEEE Pervasive Computing,12(4), 20-28.

[14] Sola, J., & Sevilla, J. 1997. Importance of input data normalization for the application of neural networks to complex industrial problems. Nuclear Science, IEEE Transactions on, 44(3), 1464-1468.

[15] Reed, R.D., and Marks II, R.J., 1999. Neural Smithing, MIT Press, Cambridge, Mass