Detection of Duplicates Among Non-structured Data From Different Data Sources

David Beauchemin

Département d'informatique et de génie logiciel,
Université Laval

david.beauchemin.5@ulaval.ca

28 july 2020

Groupe de Recherche en Apprentissage Automatique de Laval

UNIVERSITÉ LAVAL

Plan

1 Introduction

2 Databases
- *Registre des Entreprises du Québec* (REQ)
- Private Dataset

3 Similarity Between Two Entities
- Similarity Algorithm
- Machine Learning

4 Conclusion

Situation

When assessing a commercial risk, an insurer needs to gather various information about the risk. This long and complex process implies numerous questions. Thus an insurer is prompt to use an external source to help reduce the number of questions.

Situation

When assessing a commercial risk, an insurer needs to gather various information about the risk. This long and complex process implies numerous questions. Thus an insurer is prompt to use an external source to help reduce the number of questions.

Thus, we want to detect duplicate of commercial risk in another data source using as little as possible information.

Situation

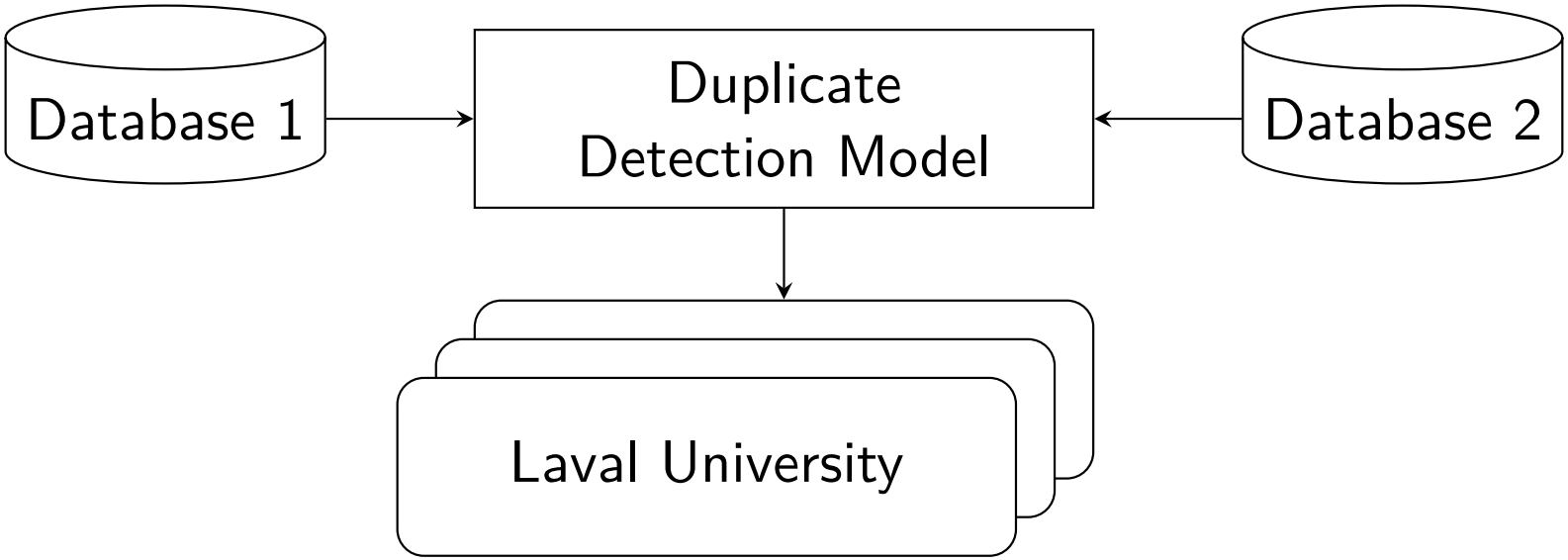
When assessing a commercial risk, an insurer needs to gather various information about the risk. This long and complex process implies numerous questions. Thus an insurer is prompt to use an external source to help reduce the number of questions.

Thus, we want to detect duplicate of commercial risk in another data source using as little as possible information.

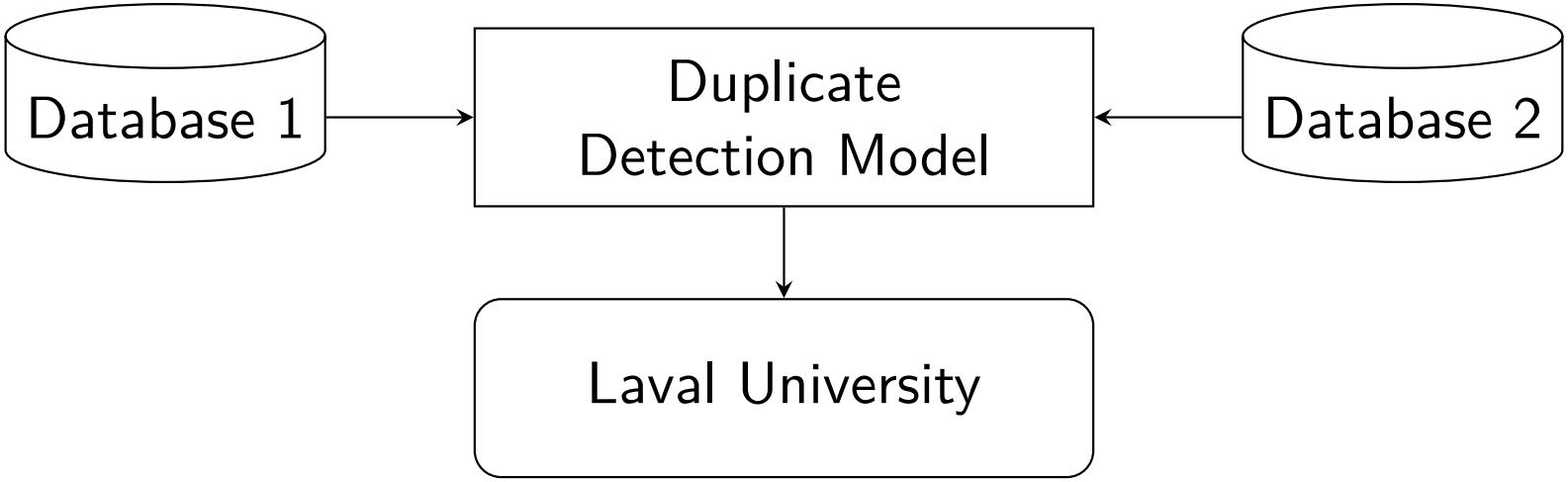
Example

David Beauchemin, the owner of "Beauchemin inc.", calls for insurance. Using minimal information, we want to retrieve as much as possible from an external source to ask him as less than the necessary number of questions.

How do we detect duplicate ?



How do we detect duplicate ?



Moreover, we consider only the top 1 document as the potential candidate.

How do we detect duplicate ?

To detect duplicate we need the following [Christen, 2012]

- databases (at least two) (section 2),
- a way to determine the similarity between two documents (section 3).

Plan

1 Introduction

2 Databases
- *Registre des Entreprises du Québec* (REQ)
- Private Dataset

3 Similarity Between Two Entities
- Similarity Algorithm
- Machine Learning

4 Conclusion

Registre des Entreprises du Québec (REQ)

- ∼3.5 millions entries
  [Registraire des entreprises du Québec, 2020]
  - Names
  - Address
  - Economic activities
  - Administrative informations

Private Dataset

- 21,444 enterprises
  - Name
  - Address
  - Economic activity

Private Dataset and the REQ

- 11,649 commercial risk are in the province of Quebec.

Private Dataset and the REQ

- 11,649 commercial risk are in the province of Quebec.
  - 1706 were annotated

Private Dataset and the REQ

- 11,649 commercial risk are in the province of Quebec.
  - 1706 were annotated
    - 1418 *(commercial risk, REQ entity)*
    - 288 *(commercial risk, None)*

Private Dataset and the REQ

- 11,649 commercial risk are in the province of Quebec.
  - 1706 were annotated
    - 1418 *(commercial risk, REQ entity)*
    - 288 *(commercial risk, None)*
- We only use the name and the address.

Name

We have used two versions of the name.

1. Normalize name (NN) : lowercase, whitespace and accent trimming.

Normalize Name

L'Université Laval ⇒ *l'universite laval*

Name

We have used two versions of the name.

1. Normalize name (NN) : lowercase, whitespace and accent trimming.

Normalize Name

L'Université Laval ⇒ *l'universite laval*

2. No stop words name (NSWN) : stop words trimming (i.e *le*, *la*, *de*) [Jurafsky and Martin, 2009].

No Stop Word Name

l'universite laval ⇒ *'universite laval*

Address

We also use two versions of the address.

1. Complete Address Normalize (NA) : same as the name.

Complete Address Normalize

2325 rue de l'Université, Québec, QC, G1V 0A6

$\Downarrow$

2325 rue de l'universite, quebec, qc, g1v 0a6

Address

We also use two versions of the address.

1. Complete Address Normalize (NA) : same as the name.

Complete Address Normalize

2325 rue de l'Université, Québec, QC, G1V 0A6

$\Downarrow$

2325 rue de l'universite, quebec, qc, g1v 0a6

2. Address components (AC) : parsed and grouped by address components [Yassine et al., 2020].

Address components

Civic Number	2325	**Unit Number**	∅
Street Name	rue de l'universite,	**Postal Code**	g1v 0a6
Orientation	∅		

Plan

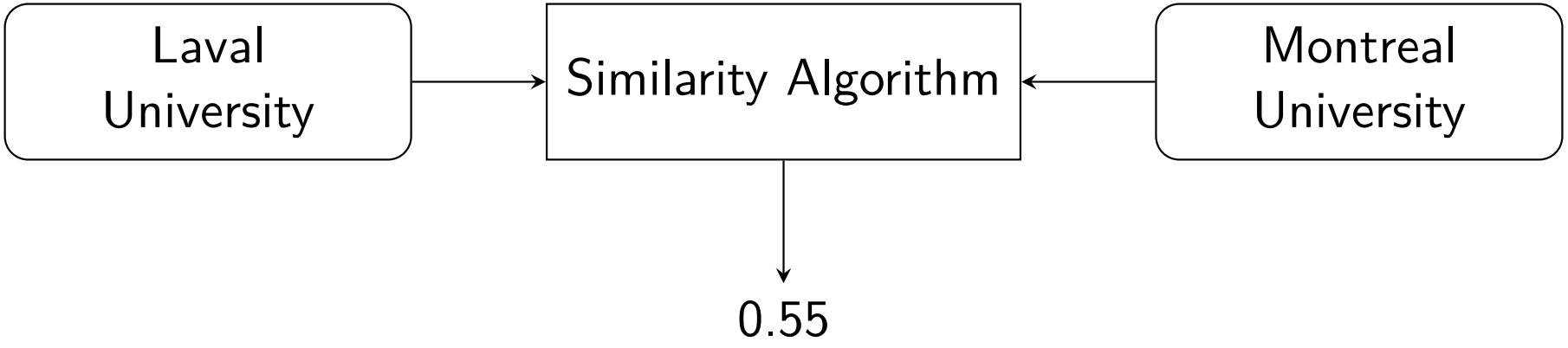
1 Introduction

2 Databases
- *Registre des Entreprises du Québec* (REQ)
- Private Dataset

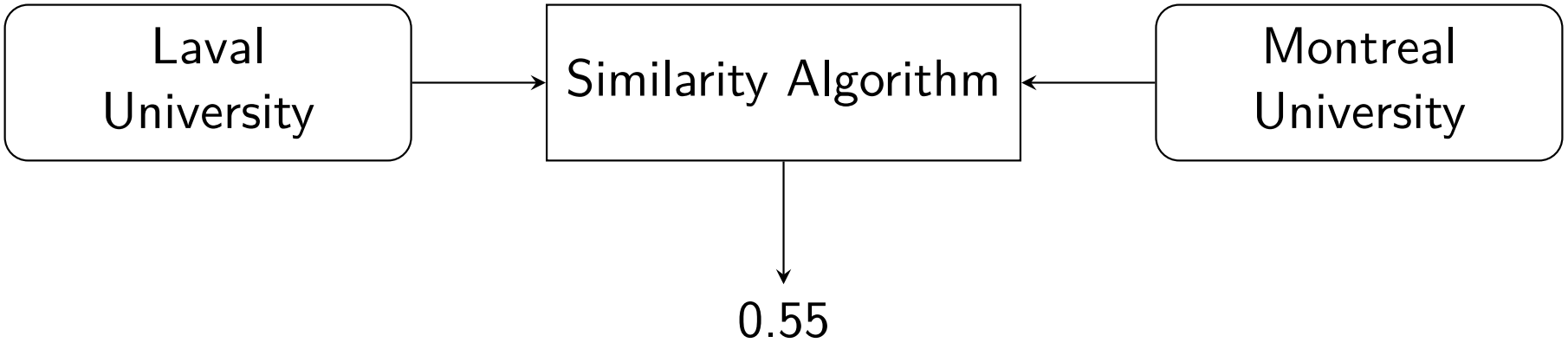
3 Similarity Between Two Entities
- Similarity Algorithm
- Machine Learning

4 Conclusion

Similarity Algorithm



Similarity Algorithm



Their similarity ranks the documents, and we select the best one as the duplicate candidate.

Similarity Algorithm

Similarities algorithms are one that measures the resemblance between two string base on the distance between their tokens.

10 different algorithms were used.

Similarity Algorithms

Jaccard Similarity

$$\text{Jaccard}(A,\ B) = \begin{cases} 0 & \text{if } |A \cap B| = 0 \text{ or } |A \cup B| = 0 \\ \frac{|A \cap B|}{|A \cup B|} & \text{otherwise} \end{cases}$$

Example

Jaccard(Laval University, Montreal University)

$$\Downarrow$$

$$\frac{|\{\text{University}\}|}{|\{\text{University, Montreal, Laval}\}|} = \frac{1}{3} = \mathbf{0{,}333}$$

Similarity Algorithms

String to String (StoS)

$$\text{StoS}(\text{A, B}) = \begin{cases} 1 & \text{if } \text{A}_i = \text{B}_j \ \forall i, j \\ 0 & \text{otherwise} \end{cases}$$

Example

StoS(Laval University, Montreal University) $\Rightarrow$ 0
StoS(Laval University, Laval University) $\Rightarrow$ 1

Similarity Algorithms

Jaro-Winkler

$$\text{Jaro-Winkler(A, B)} = \text{Jaro(A, B)} + \frac{\min(P, 4)}{10} \times (1 - \text{Jaro(A, B)})$$

Example

$$\text{Jaro-Winkler(David, Daniel)} =$$
$$\text{Jaro(David, Daniel)} + \tfrac{2}{10} \times (1 - \text{Jaro(David, Daniel)})$$
$$\Downarrow$$
$$0.7 + \tfrac{2}{10} \times 0.7 = 0.84$$

Why Similarity Algorithm ?

- Easy to used

Why Similarity Algorithm ?

- Easy to used
- No training

Why Similarity Algorithm ?

- Easy to used
- No training
- Give good results

Name Interesting Results [1]

	(%)	StoS	Jaro	Jaro-Winkler	Jaccard
NN	Accuracy	41.47	63.40	63.47	65.73
NSWN	Accuracy	44.15	64.46	65.23	**66.50**

- StoS give surprisingly good results considering the restrictive approach.

1. Positive examples only.

Name Interesting Results [1]

	(%)	StoS	Jaro	Jaro-Winkler	Jaccard
NN	Accuracy	41.47	63.40	63.47	65.73
NSWN	Accuracy	44.15	64.46	65.23	**66.50**

- StoS give surprisingly good results considering the restrictive approach.
- Removing stop words (second line) gives the best results.

1. Positive examples only.

Name Interesting Results [1]

	(%)	StoS	Jaro	Jaro-Winkler	Jaccard
NN	Accuracy	41.47	63.40	63.47	65.73
NSWN	Accuracy	44.15	64.46	65.23	**66.50**

- StoS give surprisingly good results considering the restrictive approach.
- Removing stop words (second line) gives the best results.
- The prefix similarity used by Jaro-Winkler improved results (more when using NSWN).

1. Positive examples only.

Address Interesting Results [2]

	(%)	StoS	Jaro	Jaro-Winkler	CSS
CAN	Accuracy	0.00	48.03	48.10	45.91
AC	Accuracy	13.19	51.83	51.83	**52.19**

- StoS give poor results using the normalized address due to the unomalized standard of writing (e.g., "qc" VS "(quebec)" and order of components).

2. Positive examples only.

Address Interesting Results [2]

	(%)	StoS	Jaro	Jaro-Winkler	CSS
CAN	Accuracy	0.00	48.03	48.10	45.91
AC	Accuracy	13.19	51.83	51.83	**52.19**

- StoS give poor results using the normalized address due to the unomalized standard of writing (e.g., "qc" VS "(quebec)" and order of components).
- Using the address components without considering the order of the components improved results (AC).

2. Positive examples only.

Address Interesting Results [2]

	(%)	StoS	Jaro	Jaro-Winkler	CSS
CAN	Accuracy	0.00	48.03	48.10	45.91
AC	Accuracy	13.19	51.83	51.83	**52.19**

- StoS give poor results using the normalized address due to the unomalized standard of writing (e.g., "qc" VS "(quebec)" and order of components).
- Using the address components without considering the order of the components improved results (AC).
- The local similarity used by Jaro-Winkler did not improve results this time since an address is rarely defined by is prefix.

2. Positive examples only.

Address Interesting Results [2]

	(%)	StoS	Jaro	Jaro-Winkler	CSS
CAN	Accuracy	0.00	48.03	48.10	45.91
AC	Accuracy	13.19	51.83	51.83	**52.19**

- StoS give poor results using the normalized address due to the unomalized standard of writing (e.g., "qc" VS "(quebec)" and order of components).
- Using the address components without considering the order of the components improved results (AC).
- The local similarity used by Jaro-Winkler did not improve results this time since an address is rarely defined by is prefix.
- CSS gets the best results (14% below the previous best results using Jaccard).

2. Positive examples only.

Missing Addresses

Since 16% (270) of the pair *(commercial risk, REQ entity)* are missing a address, the results are under-evaluated. For example, the CSS accuracy without those pairs is at 62.74% near 10% higher.

Those missing addresses are due to the confidentiality policy of the REQ [Registraire des entreprises du Québec, 2017].

Similarity Algorithm Summary

- Using either name or address, the similarities algorithms allows us to find the matched entity around 50% of the time. The leading approach been Jaccard using the name at near 67%.

Similarity Algorithm Summary

- Using either name or address, the similarities algorithms allows us to find the matched entity around 50% of the time. The leading approach been Jaccard using the name at near 67%.
- Using the NSWN or the AC helps improve the results over the normalized name or address.

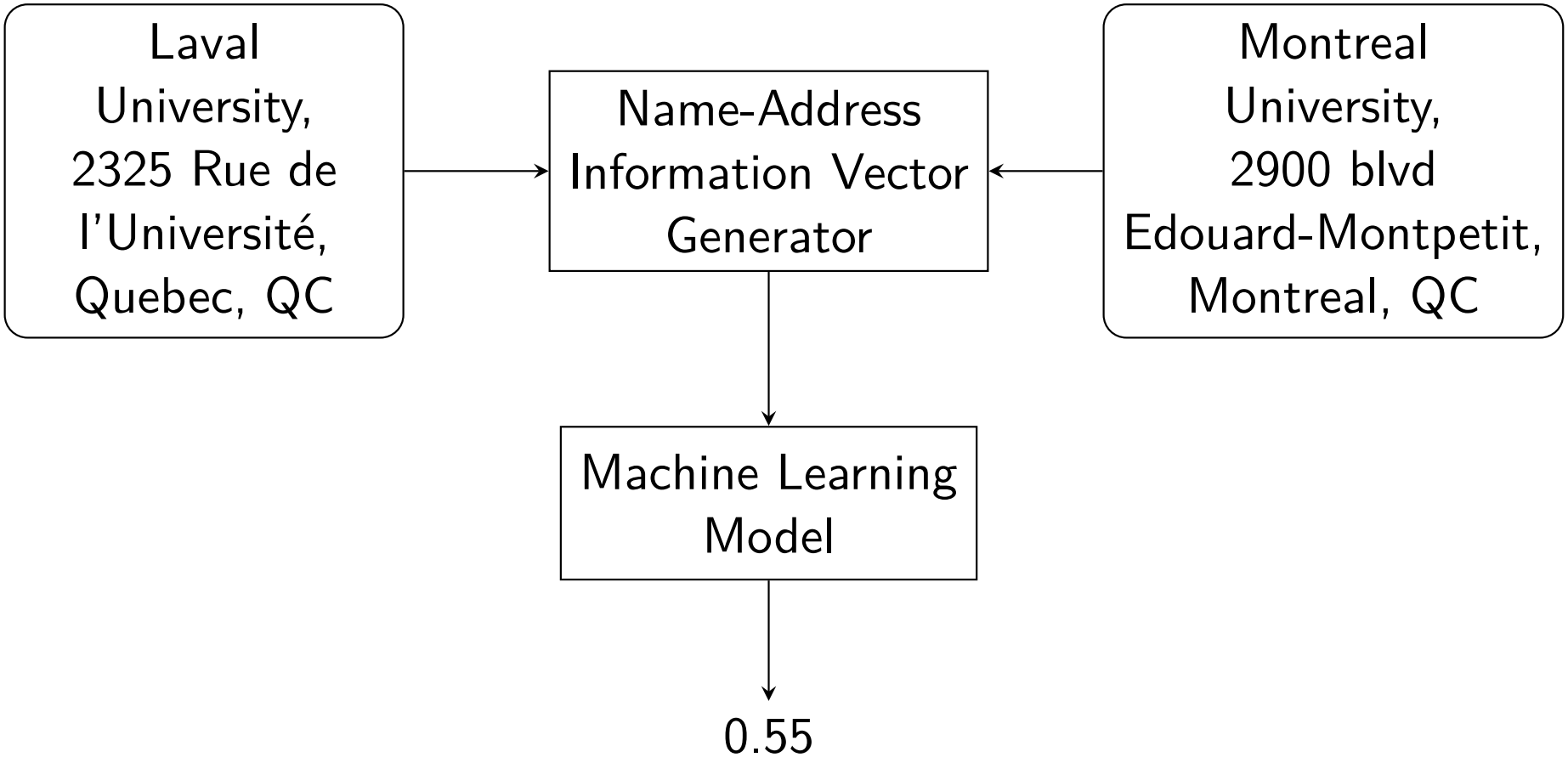
Similarity Algorithm Summary

- Using either name or address, the similarities algorithms allows us to find the matched entity around 50% of the time. The leading approach been Jaccard using the name at near 67%.
- Using the NSWN or the AC helps improve the results over the normalized name or address.
- The missing addresses pull down the results.

Similarity Algorithm Summary

- Using either name or address, the similarities algorithms allows us to find the matched entity around 50% of the time. The leading approach been Jaccard using the name at near 67%.
- Using the NSWN or the AC helps improve the results over the normalized name or address.
- The missing addresses pull down the results.
- To use positive and negative examples, we need to use a decision function. Such as a similarity threshold, were similarity below the threshold are rejected (the results are not shown).

Machine Learning Similarity Algorithm



Name-Address Information Vector Generator

We used the previous similarity algorithm to generate an information vector between two entities using the NSWN and the CA.

Example of an information vector

StoS	Levenshtein	Jaro-Winkler	LCSP	Jaccard	Cosinus	-
0.00	0.15	0.25	0.35	0.15	0.15	-
StoS	Levenshtein	Jaro	LCSP	Jaccard	Cosinus	CSS
0.00	0.16	0.55	0.15	0.45	0.37	0.48

Machine Learning Algorithm

1. Logistic regression
2. Random Forest
3. Multilayer Perceptron

Why Machine Learning ?

- Allow us to use the name and the address simultaneously
- Generalization capability

Models' Training Steps

1. Data preprocessing
2. Hyperparameters grid search
3. Model training
4. Evaluation of the trained model for the duplicate detection task

Data preprocessing

- 80-20 splitting of the data (1706 annoted exemples).

Data preprocessing

- 80-20 splitting of the data (1706 annoted exemples).
- From the train dataset (80%), we randomly match the negatives pair *(commercial risk, None)* (241 examples) with a REQ entity.

Data preprocessing

- 80-20 splitting of the data (1706 annoted exemples).
- From the train dataset (80%), we randomly match the negatives pair *(commercial risk, None)* (241 examples) with a REQ entity.
- To balance the dataset (1364 positives vs. 241 negatives), we randomly select the first name and address of a REQ entity to create a fake commercial risk and randomly pair it with another REQ entity. Resulting in a training dataset of 2246 *(commercial risk, REQ entity)* pair.

Data preprocessing

- 80-20 splitting of the data (1706 annoted exemples).
- From the train dataset (80%), we randomly match the negatives pair *(commercial risk, None)* (241 examples) with a REQ entity.
- To balance the dataset (1364 positives vs. 241 negatives), we randomly select the first name and address of a REQ entity to create a fake commercial risk and randomly pair it with another REQ entity. Resulting in a training dataset of 2246 *(commercial risk, REQ entity)* pair.
- We fit a standard scaler on the training dataset. With that standard scaler, we applied a transformation over all the vectors (train and test).

Hyperparameters Grid Search & Model Training

- A grid search for the logistic regression (C and tolerance) and the random forest (number of estimators).
- A random search for the multilayer perceptron (number of layers and neurons and the tolerance).
- Cross-validation approach using a 5-folds.

After the grid search, we retrain using the best parameters.

Evaluation

We evaluate our three trained models against the best configuration, Jaccard using the name, but reevaluated with the validation dataset (20% of the 1706 annotated examples).

We evaluated the algorithm with positives and negatives examples (for recall and precision). The decision function is a similarity threshold where a similarity below a numerical threshold (e.g., 0.7) is rejected.

We aim to maximize the recall since our objective is to detect the more duplicate as possible since we can, later on, validate the duplicate manually.

Results

(%)	Logistic Regression	Random Forest	Multilayer perceptron	Jaccard
Precision	**89,77**	81,06	87,55	81,78
Recall	66,67	73,54	**79,73**	72,51
Accuracy	64,91	62,87	**73,10**	62,87

- The random forest and the multilayer perceptron achieve better recall than Jaccard. The best being the perceptron with near 80% recall.

Results

(%)	Logistic Regression	Random Forest	Multilayer perceptron	Jaccard
Precision	**89,77**	81,06	87,55	81,78
Recall	66,67	73,54	**79,73**	72,51
Accuracy	64,91	62,87	**73,10**	62,87

- The random forest and the multilayer perceptron achieve better recall than Jaccard. The best being the perceptron with near 80% recall.
- The logistic regression achieves the lowest result even if the Jaccard similarity is used in the generation of the information vector.

Example of an Error

Commercial Risk	Entity
construction alain cloutier inc. 1030 rue de l'ardoise sherbrooke j1c 0j6	construction steeve arbourd inc. 2-1822 rue notre-dame, l'ancienne-lorette, g2e 3c7

The following error was only generated with the logistic regression model. The algorithm matched the two with a similarity of 0.99934, even if the annotated duplicate appear with the same name and a slightly different address (similarity of 0.998).

Improving the Results

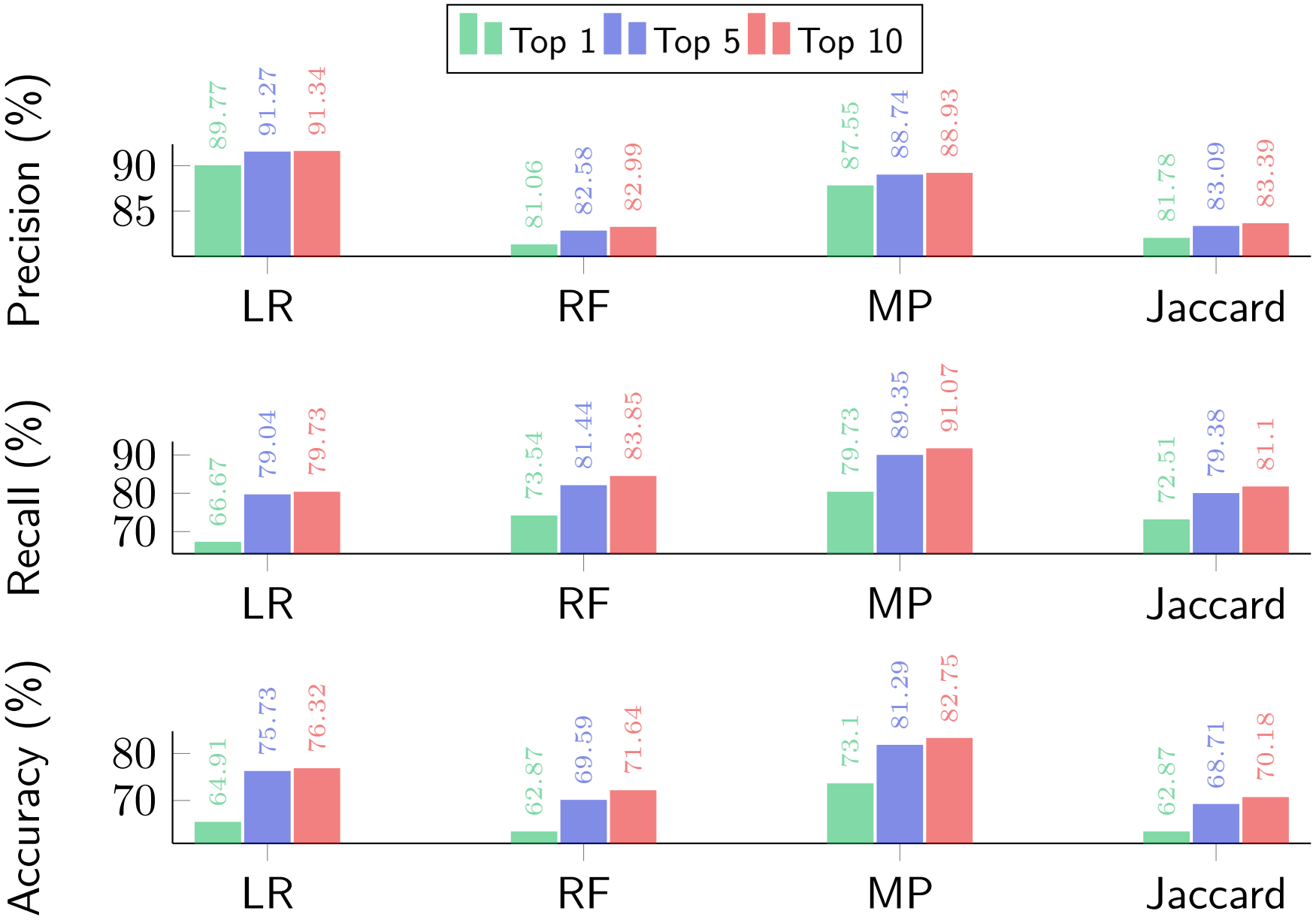
The previous error highlights a problem of our approach ; so far, we have tried to match with **the** most similar, and the generated similarity are close to each other, making it restrictive to use only the best similarity.

Since our approach aims to prefill the information of an insurance application, we can return more than one and use a human validation to select the best one from N possibilities.

N Most Similar

We consider a matching is good when the pair *(commercial risk, REQ entity)* is included in the N most similar.

Results


Top 1
Top 5
Top 10
Precision (%)
90
85
89.77
91.27
91.34
81.06
82.58
82.99
87.55
88.74
88.93
81.78
83.09
83.39
LR
RF
MP
Jaccard
Recall (%)
90
80
70
66.67
79.04
79.73
73.54
81.44
83.85
79.73
89.35
91.07
72.51
79.38
81.1
LR
RF
MP
Jaccard
Accuracy (%)
80
70
64.91
75.73
76.32
62.87
69.59
71.64
73.1
81.29
82.75
62.87
68.71
70.18
LR
RF
MP
Jaccard

Results

- Using a top N approach greatly improved the results.
- Using $N = 10$, we can achieve a near max recall at 91% with the multilayer perceptron (max of 93% with the indexing).

Inference Times

(second)	Logistic Regression	Random Forest	Multilayer Perceptron	Jaccard
Time	1,32	1,74	1,34	0,25

Machine Learning Algorithm Summary

- The used of name and address simultaneously with a machine learning algorithm improved the results.

Machine Learning Algorithm Summary

- The used of name and address simultaneously with a machine learning algorithm improved the results.
- Using a top N approach helps achieve better results when N is greater than 1.

Machine Learning Algorithm Summary

- The used of name and address simultaneously with a machine learning algorithm improved the results.
- Using a top N approach helps achieve better results when N is greater than 1.
- Inference times (of machine learning models) are similar to using only a similarity algorithm.

Plan

1 Introduction

2 Databases
- *Registre des Entreprises du Québec* (REQ)
- Private Dataset

3 Similarity Between Two Entities
- Similarity Algorithm
- Machine Learning

4 Conclusion

Conclusion

- We have shown that using a similarity algorithm can achieve good results.
- Uses of machine learning algorithm (such as multilayer perceptron) can achieve greater results.
- Using a N most similar approach, where N is greater than one, help improved the results, achieving almost the max recall value.

Future Works

- Word embeddings [Mikolov et al., 2013, Pennington et al., 2014, Wu et al., 2017, Foxcroft et al., 2019, Singh et al., 2019]
- Siamese Network [Godbole et al., 2018, Imtiaz et al., 2020]
- Uses of spatial data [Sehgal et al., 2006]
- Removal of more specitif stop words using a TF-IDF approach [Sammut and Webb, 2010]

Acknowledgment

- This research was supported by the Natural Sciences and Engineering Research Council of Canada (IRCPJ 529529-17) and a Canadian insurance company.
- Luc Lamontagne for his mentorship.

References I

Christen, P. (2012).
Data matching : concepts and techniques for record linkage, entity resolution, and duplicate detection.
Springer Science & Business Media.

Foxcroft, J., d'Alessandro, A., and Antonie, L. (2019).
Name2vec : Personal names embeddings.
In *Advances in Artificial Intelligence*, pages 505–510.

Godbole, A., Dalmia, A., and Sahu, S. K. (2018).
Siamese neural networks with random forest for detecting duplicate question pairs.

Imtiaz, Z., Umer, M., Ahmad, M., Ullah, S., Choi, G. S., and Mehmood, A. (2020).
Duplicate Questions Pair Detection Using Siamese MaLSTM.
IEEE Access, 8 :21932–21942.

References II

Jurafsky, D. and Martin, J. H. (2009).
Part-of-speech tagging.
In *Speech and Language Processing (2nd Edition)*, chapter 8.
Prentice-Hall, Inc.

Mikolov, T., Chen, K., Corrado, G., and Dean, J. (2013).
Efficient Estimation of Word Representations in Vector Space.

Pennington, J., Socher, R., and Manning, C. (2014).
GloVe : Global vectors for word representation.
In *Proceedings of the Conference on Empirical Methods in Natural Language Processing*, pages 1532–1543.

Registraire des entreprises du Québec (2017).
Banque de données publique sur les entreprises au Québec - Guide d'intégration fonctionnelle.
Québec.

References III

Registraire des entreprises du Québec (2020).
Le registre et son contenu.
http://www.registreentreprises.gouv.qc.ca/fr/a_propos/registre/.

Sammut, C. and Webb, G. I., editors (2010).
TF–IDF, pages 986–987.
Springer US.

Sehgal, V., Getoor, L., and Viechnicki, P. D. (2006).
Entity resolution in geospatial data integration.
In *Proceedings of the 14th Annual ACM International Symposium on Advances in Geographic Information Systems*, page 83–90.
Association for Computing Machinery.

Singh, L., Singh, S., Arora, S., and Borar, S. (2019).
One embedding to do them all.

References IV

Wu, L., Fisch, A., Chopra, S., Adams, K., Bordes, A., and Weston, J. (2017).
Starspace : Embed all the things !

Yassine, M., Beauchemin, D., Laviolette, F., and Lamontagne, L. (2020).
Leveraging subword embeddings for multinational address parsing.