Markov Networks as Prior Distribution for Multi-Label Classification (in Wikipedia)

#### By: Alon Wolf (318336930) and Aviv Hadar (208948885)

*code:* [*https://github.com/AlonWolfi/wikipedia-summary*](https://github.com/AlonWolfi/wikipedia-summary)

# תקציר - Abstract

פעמים רבות, כאשר נתקלים בבעיית *MultiLabel*, נאלצים להשתמשים במסווגים לפי שיטת *OneVsRest*. שיטה זו לוקה בחסר מכיוון שאינה מתייחסת לקשרים השונים בין המחלקות. למשל, בזיהויי אובייקטים בתמונה - כמובן שהגיוני יותר שספינה ועוגן יופיעו ביחד בתמונה מאשר ספינה והר או מכונית ועוגן. מודל *OneVsRest* יתעלם מקשרים כגון אלו בין המחלקות.

במאמר זה, בחנו שימוש בואריציה על מודל הסתברותי מסוג רשת מרקובית כ-prior על הקשרים בין המחלקות השונות, וזאת במטרה לשפר חיזויים של מסווג בבעיית *MultiLabel*.

אנו השתמשנו במודל המרקובי ובהסתברויות שחזרו מהמסווג ויצרנו חיזויים חדשים על המחלקות בעזרת ואריציה על אלגוריתם *Belief Propogation*. השתמשנו בואריציות שונות של אלגוריתם *Chow Liu* לבחירת מבנה רשת זו.

# רקע תאורטי

## מודלים מרקוביים

כפי שציינו, במאמר נעסוק בשיפור ביצועי אלגוריתם בבעיית *MultiLabel* בעזרת יצירת prior על התיוגים בין מחלקות. זאת באמצעות מודל מרקובי מעל להתפלגות הלייבילים שנייצר עפ"י אלגוריתם *Chow Liu*[[1]](#footnote-1) (כפי שיפורט בהמשך).

עבור דוגמא ספיציפית, נסמן את התיוגים האמיתיים עבור המחלקה:

אלגוריתם *Chow Liu*, כפי שנראה בהמשך, מחזיר עץ. מכיוון שבעץ אין קליקות בגודל גדול מ-2, רשת מרקובית עם מבנה כנ"ל מעל המחלקות תהיה - *Pairwise Markov Network*.

במקרה הזה מתקיים שזו הנוסחה ל-*Inference*:

מכיוון שמבנה הרשת הוא עץ (כפי שמתקבל לפי האלגוריתם) נקבל שהמשקלים האופטימליים לרשת הם:

כאשר מתקיים:

|  |  |
| --- | --- |
|  | (3) |

## Belief Propagation

נזכור שלכל *Classifier* נרצה לחשב את ה-*Marginals* כדי לדעת את הסבירות של כל לייבל להיות חיובי. כלומר נרצה לחשב את , כאשר , אלו הפיצ'רים של הדוגמה.

ניתן לחשב Marginals של מודל מרקובי עם מבנה של עץ בעזרת שיטת ה-Belief Propagation בצורה הבאה:

כאשר מתקיים:

משוואה (4) בעצם נותנת לנו את ההסתברות לקיומו של תיוג מסוים לפי **התפלגות התיוגים בלבד**, זאת מפני שהמודל הנ"ל ממשקל את כל ערכי באופן זהה ו- מושפע מ- בלבד (בהינתן הצבת המשקלים האופטימליים). כלומר המודל מתעלם כרגע מהסבירות לתיוגים שונים בהינתן מאפייני הדוגמא, שזו יכולה באופן משמעותי להשתנות בין דוגמאות שונות בהתאם לערכי הפיצ'רים.

המטרה שלנו בפרויקט הייתה לבצע תיקון לחיזויים של המודל ע"י שימוש ב-prior שישלב את התפלגות הלייבלים עם פרדיקציות ה-classifier.

איך ביצענו זאת? יצרנו מודל מרקובי מעל הלייבלים () באופן הנ"ל, ובנוסחה לחישוב ה-Marginals (בשיטת ה-belief propagation) משקלנו את ערכי בכל הודעה לפי החיזויים של המודל - .

כאשר מתקיים:

כלומר, מה שאנו רוצים לעשות הוא לשלב את הסבירות לקיומו של לייבל בהינתן הפיצ'רים () עבור מחלקה יחידה עם ההסתברות שנובעת מהתפלגות התיוגים (), במטרה לשפר את ביצועי המודל.

*הנוסחה המשולבת היא להלן:*

|  |  |
| --- | --- |
|  | (6) |

*חשוב לציין ש- אינה פונקציית הסתברות שכן נדרש לנרמל אותה:*

*זהו החיזוי עבור כל תיוג שמוחזרת אחרי התיקון שלנו לפי ה-prior.*

## אלגוריתם צ'או ליאו

*נחזור לאלגוריתם Chow Liu. כאמור השתמשנו בו בשביל לבנות את המודל המרקובי לפי ה-prior. השתמשנו בו באופן הבא:*

1. *הגדרנו גרף קשרים בין המחלקות עם המשקלים הבאים:*
2. *לאחר מכן מצאנו עץ פורש מקסימאלי לגרף*
3. *והעץ שקיבלנו שימש אותנו לבניית המודל המרקובי*

*השתמשנו בואריציות של האלגוריתם כך שיתאימו יותר לשיפור הביצועים והשוואנו אותם לתהליך המקורי.*



*איור 1 – ערך ויקיפדיה של אלטון ג'ון. באדום – הערך (טקסט), בכחול – עובדות מהירות. מטרת הפרויקט היא לחזות את החלק הכחול בהינתן האדום.*

# מידע (דאטה)

## הגדרת הבעיה

את המידע חילצנו מאתר ויקיפדיה. השתמשנו בספריות *wikipediaapi*, *wptools* על מנת לחלץ את הטקסט וה-*infobox* של ערכי הויקיפדיה (דוגמה לערך ניתן לראות ב*איור 1*).

לקחנו את הטקסט כמשתנה הבלתי-תלוי ואת השאלות של ה-*infobox* כמשתנה התלוי. כלומר למצוא את הקטגוריות ב-infobox (הם יקראו שאלות במאמר זה), בהינתן כל הטקסט שבערך.

נשאלת השאלה באילו ערכי ויקיפדיה להשתמש – לא היינו יכולים לקחת את כל ערכי ויקיפדיה כיוון שמספרם גדול מדי ודגימה רנדומלית מתוך ויקיפדיה אינה ממשית (וכן צריך דגימה גדולה מאוד בשביל שתספיק תייצג את כלל הערכים). על כן, דגמנו את הערכים מ-subset של ערכי ויקיפדיה והוא "מוזיקאים"[[2]](#footnote-2) (בכדי שהערכים ייבחרו על ידי התפלגות יחידה ומוגדרת היטב).

## Preprocess

את הטקסט שבערכי הויקיפדיה העברנו עיבוד פר מילה כך שלכל מילה ייצרנו ממד חדש על האם הטקסט הכיל את המילה (ללא התייחסות לסדר). בנוסף ביצענו פעולות preprocess נוספות, דוגמה לתהליך זה עבור המילה *'Birth\_date'* ניתן לראות ב*איור 2*. לכל מילה הכנסנו ציון לפי אלגוריתם TF-IDF כך שככל שמילה מופיעה יותר בטקסט כך היא מקבלת ציון גבוה יותר, מנורמל לפי מס' הפעמים שהמילה הופיעה בכלל הטקסטים.

בנוסף התעלמנו ממילים שחזרו במס' בודד של טקסטים כדי לצמצם משמעותית את כמות הפיצ'רים ולא לפגוע בביצועים (וגם זה יקל על המודל להימנע מ-overfit).

*איור 2 – עיבוד המילה על מילה "Birth\_date". ניתן לראות שכל חלק ...*

Tokanization

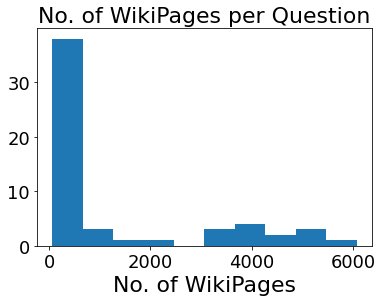
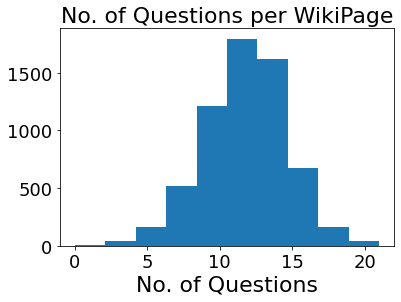
*'Birth\_date'*

Stop Words Deletion

*'Birthdate'*

Stemming

*'Birthdat'*



*איור 3 – היסטוגרמות של תיוגים ומחלקות. כמה שאלות (תיוגים) יש לכל דוגמא וכמה דוגמאות יש לכל שאלה (מחלקה)*

השתמשנו בpreprocess - stemming גם על השאלות, והסיבה לכך היא שבשאלות היו עיוותים של מילים שמתארות את אותו הדבר - כן לדוג' *'Birth\_date', 'birth\_date', 'birthdate'* הופיעו בערכים שונים וכמובן שהכוונה בהם הייתה זהה. לאחר העיבוד שלנו הפכו למילה אחת *'birthdat'*.

כמו כן, מתוך השאלות גם כן סיננו שאלות לא נפוצות (מופיעות עבור מס' בודד של ערכים) שמספר מופען קטן ולכן אינן מייצגות את התפלגות המחלקה.

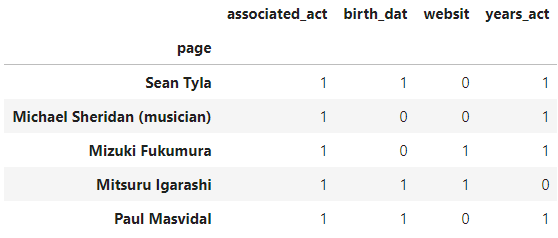
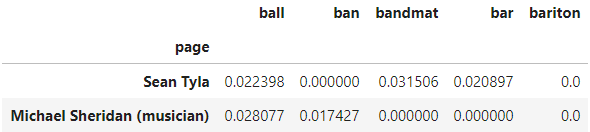
כפי שניתן לראות מהדוג', הבעיה הינה *MultiLabel* כיוון שלכל ערך ישנו מס' משתנה של שאלות.

את התפלגות מס' השאלות לכל ערך ניתן לראות באיור מס' 3. כפי שניתן לראות מהאיור, הופעת התיוגים מגוונת ואיננה טריוואלית – כלומר יש שאלות שמופיעות עבור עשרות דפים (כגון *'height'*) יש שאלות שמופיעות עבור מאות דפים (כגון *'record'*), ויש שאלות שמופיעות עבור רוב הדפים (כגון *'name'*).

## סיכום המידע

סה"כ קיבלנו 6221 דוגמאות עם 4328 פיצ'רים ו-56 סוגי מחלקות.

דוגמא למידע שקיבלנו ניתן לראות ב*טבלה 1*. ניתן לראות דוגמאות לפיצ'רים שקיבלנו וכן דוגמאות לשאלות עבור 2 ערכים – 'Sean Tyla'[[3]](#footnote-3), 'Michael Sheridan'[[4]](#footnote-4).



*טבלה 1 – דוגמא למידע שקיבלנו עבור 2 דפי ויקיפדיה. באיור העליון דוג' לשאלות (תיוגים) עבור הערכים. באיור התחתון דוג' לערכי פיצ'רים עבור הערכים.*

# תהליך

התחלנו בלייצר מודל לכל מחלקה (שאלה). המודלים פעלו בשיטת *OneVsRest* – כלומר, יצרנו מודל לכל מחלקה בנפרד. עשינו זאת כיוון שזהו common practice שמשתמשים בו כשרוצים למדל בעיות של *MultiLabel* (כיוון שיכול להיות יותר מתיוג אחד עבור כל מחלקה ולכן מודל מבוסס "*softmax*" אינו מתאים) . נשים לב שהמודל לא מתחשב בקשרים בין המחלקות כיוון שהוא מנתח כל מחלקה בנפרד.

את תיקון החיזויים ע"י ה-prior () חישבנו כפי שתיארנו בחלק התאורטי.

זאת בשלבים הבאים:

1. את חישבנו לפי סט האימון בלבד בשביל לא לייצר *Target Leakage* בבחינה של המודל.
2. *בחרנו את מבנה העץ של המודל המרקובי ע"י אלגוריתם Chow Liu (או ואריציה של האלגוריתם).*
3. השתמשנו בעץ וב- בשביל לחשב את על חיזוי המודל עפ"י האמור במשוואות (6) ו-(7) בעזרת *Belief Propagation* על המודל המרקובי.

*בהמשך בחנו את המודל תחת מבני עץ שנבנו בצורות שונות מזו, כפי שיוצג בחלק הבא.*

השתמשנו בסוגי מודלים שונים על מנת לבחון את עצמנו במס' תרחישים. אימנו את המודל על משקלי מחלקות מאוזנים, זאת על מנת שהחיזויים של המודל יהיו הסתברות.

\* יש לשים לב כי לפי הנוסחא ב-, משקול ההודעות החוזרות מתבצע לפי פרדיקציות המודל ו**מניח כי פרדיקציות אלו הן אכן הסתברויות.** לכן בכדי שייצגו פונקציות הסתברות, אנחנו ממליצים לנרמל את משקלי המחלקות באימון המודל.

# תוצאות

## אופן בחינת מודל

השוואנו את תוצאות המודלים לפי 2 מדדים*:*

*ROC AUC – במטרה לבחון את הדירוג שהחזיר המודל. כלומר במטרה לראות האם הסיכוי של תיוג חיובי להופיע לפני דוגמא שלילית גדל או קטן. כי ROC AUC מושפע הסדר בו המודל מדרג את הדוגמאות ולא מהציונים עצמם.*

*F1 – במטרה לבחון את המודל כמסווג. כלומר לראות האם החלוקה האופטימלית של המודל כמסווג בין חיוביים לשליליים השתפרה.*

כיוון ש- דורש *להקשיח את הפרדיקציות בחרנו את סף הציון (ה-decision boundary להלן ) אשר ממקסם את הציון על הסט הואלידציה (כך ש- ) (זאת כיוון שהבחנו שסף הציון האופטימלי עבור, במקרים רבים, אינו בהכרח 0.5 ולכן בחרנו אותו בשיטה זו ואנו ממליצים למי לבחור סף בצורה זו למי שמשתמש בשיטות דומות). למען הסר ספק, ביצענו את התהליך הזה עבור כל מודל, לפני ואחרי השימוש בprior.*

*כיוון שיש לנו מס' מחלקות, מיצענו בין המדדים בשיטת 'macro'. השתמשנו בשיטת 'macro' כיוון שיש מס' מחלקות שמרביתן חיוביות ויש מס' מחלקות שמרביתן שליליות, ומצאנו לנכון לבחון אותן ללא תלות במס' החיוביים.*

השתמשנו בשיטת *K-Fold Cross Validation* כאשר k=5. הציונים המוצגים בתוצאות הינם ממוצע החלוקות השונות. דייקנו את התוצאות עד לסטיית תקן אחת של התוצאות מה-fold-ים השונים (4 ספרות אחרי הנקודה).

## חלק I - מודל מרקובי

ראשית, בדקנו את הציון של מודל OneVsRest עבור 2 מודלי בסיס שונים (*SVM, LightGBM) (אחד מבוסס עצים ואחד ליניארי) והשוונו אותם לציון של המודל לאחר הפעלת ה-prior. ניתן לראות את התוצאות ב*טבלה 2*.*

|  |  |  |
| --- | --- | --- |
| LGB | SVM |  |
| 0.7084 | 0.7095 | OneVsRest |
| **0.7175** | **0.7109** | OneVsRest and Prior |
| 0.0091 | 0.0014 | Diffrence |

|  |  |  |
| --- | --- | --- |
| LGB | SVM |  |
| 0.9059 | 0.8981 | OneVsRest |
| **0.9068** | **0.9024** | OneVsRest and Prior |
| 0.0009 | 0.0043 | Diffrence |

*טבלה 2 – תוצאות עבור חלק 1*

ניתן לראות כי התיקון לפי ה-prior משפר את התוצאות עבור 2 המודלים. אמנם בעשיריות אחוזים בלבד, אך בעקבות גודל סטיית התקן בין ה-fold-ים השונים ניתן לומר שהשיפור מובהק סטטיסטית.

## חלק II - סוגי עצים

לאחר מכן, רצינו לבדוק כיצד מבנים שונים של מודלים מרקוביים משפיעים על טיב התוצאות.

בנינו מודלים שונים ע"י ואריציות על אלגו' *Chow Liu*.

הסתכלנו על:

1. עץ רנדומלי – בדיקת שפיות שנועדה לבדוק שהשיפורים אכן מגיעים מטיב האלגוריתם והמודל המרקובי.
2. *M.I.* – אלגוריתם *Chow Liu* הקלאסי. העץ נוצר לפי משקלים בין מחלקות המבוססים ב-MI בין התיוגים של המחלקות. (*משוואה (8)*)
3. *'Semi-Positive' M.I.* – נשים לב שנוסחת MI מורכבת מ-4 חלקים מארבעת האפשרויות לערכים של , שכן MI מתארת כמה "אינפורמציה" תיוג מכיל על תיוג ולהיפך, אמנם, לא בהכרח 4 האפשרויות משפיעות על תיקון הפרדיקציות כפי שהשתמשנו בו על העץ. כך לדוגמא, 2 לייבלים שלא מופיעים כמעט אף פעם כחיובים, , בעלי MI גבוה. כביכול, בעלי אינפורמציה משותפת גבוהה, כיוון שאם הראשון אינו חיובי סימן שגם השני ככל הנראה לא חיובי. אמנם, הם לא אמורים להשפיע על הפרדיקציות אחד של השני, ה-MI ביניהם גבוה אמנם הראשון לא אומר על השני הרבה על איכות החיזוי ולהיפך.

על כן, ניסינו לתקן את המשקלים כך שיתחשבו רק ב-.

1. *'Positive' M.I.* – כעת ניסינו להמשיך קו מחשבה זה אף מעבר ולהתייחס רק למקרים בהם כאשר .

תוצאות לחלק זה ניתן לראות ב*טבלה 3*.

|  |  |  |
| --- | --- | --- |
| LGB | SVM |  |
| 0.7084 | 0.7095 | baseline |
| 0.7086 | 0.7094 | Random |
| 0.7175 | **0.7109** | M.I. |
| **0.7177** | 0.7099 | 'Positive' M.I. |
| 0.7145 | 0.7084 | 'Semi-Positive' M.I. |

|  |  |  |
| --- | --- | --- |
| LGB | SVM |  |
| 0.9059 | 0.8981 | baseline |
| 0.9068 | 0.8996 | Random |
| **0.9097** | 0.9024 | M.I. |
| 0.9094 | **0.9041** | 'Positive' M.I. |
| 0.9085 | 0.8999 | 'Semi-Positive' M.I. |

*טבלה 3 – תוצאות עבור חלק 2*

נשים לב כי העץ הרנדומלי לא משפר את תוצאות המודל המקורי. זאת הבחנה חשובה כיוון שהיא תומכת בכך שיש חשיבות לבחירת מבנה המודל המרקובי על המחלקות. חשוב לציין שאף מבנה רנדומלי לא פוגע בתוצאות המודל המקורי לאחר התיקון עפ"י ה-*prior*.

ניתן לראות כי שלושת העצים אכן משפרים את תוצאות המודל (במס' עשיריות האחוז ולכן השיפור מובהק סטטיסטית). ניתן לראות כי התוצאות הטובות ביותר קיבלנו עבור אלגוריתם *Chow Liu* הסטנדרטי (*M.I.*) והואריציה שלנו המבוססת על קשרים "חיובים" בלבד בין המחלקות כפי שהיא מתוארת לעיל (*'Positive' M.I.*).

## חלק III - הרכבה של מודלים

נשים לב שהפעלת התיקון עפ"י ה-prior מהווה אופרטור אשר מקבל חיזויים עפ"י מודל מסוים ומחזיר חיזויים חדשים. כפי שראינו אחרי הפעלת ה-prior קיבלנו חיזויים טובים יותר מהחיזויים המקוריים (מופיע בחלקים I, II). על כן, שיערנו כי אם נפעיל את התיקון על החיזויים המשופרים נקבל שיפור נוסף בתוצאות (כיוון שהתיקון מושפע מהחיזויים עצמם).

ניסינו את תהליך הרכבת התיקונים עם אותו האופרטור (*M.I. -> M.I*.) ועבור הרכבה של אופרטורים שונים (*M.I. -> 'Positive' M.I.*) מתוך הנחה שהם מכילים אינפורמציה שונה על הקשרים בין המחלקות.

הבחנו שהמידע שחוזר מהחיזויים לאחר התיקון אינו מתפלג עפ"י פונ' הסתברות כיוון שסף הציון האידיאלי הופיע במרבית המקרים סמוך לציון 1.0 או ציון 0.0 (ולא 0.5 כפי שהיינו מצפים מפונקציית הסתברות). על כן, נרמלנו את הפרדיקציות לפי סף הציון על ידי קליברציה ליניארית של החיזויים (כפי שמתואר ב*נספח I*). זאת בתקווה שלאחר קליברציה של החיזויים, אלו יהוו בצורה מהיימנה יותר את ההסתברות לקיום הלייבל.

תוצאות ניתן לראות ב*טבלה 4*.

|  |  |  |
| --- | --- | --- |
| LGB with calibration | LGB |  |
| - | 0.7084 | baseline |
| - | 0.7175 | M.I. |
| 0.7196 | 0.7156 | M.I. -> M.I. |
| **0.7220** | 0.7172 | M.I. -> 'Positive' M.I. |

|  |  |  |
| --- | --- | --- |
| LGB with calibration | LGB |  |
| - | 0.9059 | baseline |
| - | 0.9097 | M.I. |
| 0.9082 | 0.9076 | M.I. -> M.I. |
| **0.9101** | 0.9095 | M.I. -> 'Positive' M.I. |

*טבלה 4 – תוצאות עבור חלק 3*

ניתן לראות כי הרכבת המודלים לא משפרת הביצועים (ואף עלולה לפגוע בהם), ללא הקליברציה של החיזויים. אמנם, אחרי הפעלת הקליברציה אנחנו מקבלים שהתוצאות משתפרות עבור 2 סוגי ההרכבות, בסדר גודל דומה לשיפור לאחר התיקון הראשוני של ה-prior.

כמו כן, ניתן לראות שהתוצאות הטובות ביותר מתקבלות כאשר משתמשים במודלים מרקוביים שונים. להערכתנו, זאת מכיוון שכל מודל מכיל מידע שונה על הקשרים בין המחלקות ודרך הרכבה של מודלים שונים איחדנו בין המידע שבין שניהם בצורה הטובה ביותר.

# סיכום ומסקנות

במאמר זה, בחנו שימוש בואריציה על מודל הסתברותי מסוג רשת מרקובית כ-prior על הקשרים בין המחלקות השונות, וזאת במטרה לשפר חיזויים של מסווג בבעיית *MultiLabel*.

השתמשנו במודל המרקובי ובהסתברויות שחזרו מהמסווג, ובעזרת משוואות (6), (7), יצרנו חיזויים חדשים על המחלקות. השתמשנו בואריציות שונות של אלגוריתם *Chow Liu* לבחירת מבנה רשת זו.

בחנו את החיזויים החדשים לפי ה-*prior* עבור 2 מודלים שונים (*SVM, LightGBM*) מבוססי *OneVsRest*. השוונו את התוצאות לפני ואחרי התיקון שיצרנו וקיבלנו כי אכן יש שיפור מובהק סטטיסטית בביצועים של המודל.

לבסוף, ניסינו להרכיב את התיקונים עפ"י ה-*prior* עבור מבנים שונים של הרשת המרקובית וקיבלנו שיפור מובהק סטטיסטית, זאת רק לאחר שימוש בקליברציית הסתברויות על החיזויים שחזרו לאחר תיקון החיזויים לפי ה-*prior*.

מקרה הבוחן עליו בחרנו לבחון את כך הוא חיזוי פרטי מידע שהופיעו ב-*infobox* עבור ערכי *Wikipedia* שונים.

לסיכום, הוכחנו את טיב השימוש במודל מרקובי כ-*prior* לשיפור הביצועים של מסווג בבעיות מסוג *MultiLabel*.

# נספחים

## נספח I

עבור מחלקה ספיציפית , נגדיר את החיזויים עבור כלל התצפיות כוקטור וכן את סף הציון המקסימלי באופן זהה לזה שהגדרנו במאמר.

על כן, ניעזר בקליברציה ש"מזיזה" את סף הציון ל-0.5 באופן הבא:

1. <https://en.wikipedia.org/wiki/Chow%E2%80%93Liu_tree> [↑](#footnote-ref-1)
2. <https://en.wikipedia.org/wiki/Category:Musicians> [↑](#footnote-ref-2)
3. <https://en.wikipedia.org/wiki/Sean_Tyla> [↑](#footnote-ref-3)
4. <https://en.wikipedia.org/wiki/Michael_Sheridan_(musician)> [↑](#footnote-ref-4)