

המרכז האקדמי רופין

בית-הספר להנדסה

המחלקה להנדסת חשמל ומחשבים

ספר פרויקט לשנת תש"פ

נושא הפרויקט:

Deep Learning for Malware Classification

מבצעים:

עדן אביטן	ת.ז 203286992
רועי אספורטס	ת.ז 205718083
תומר עפרוני	ת.ז 205711682

מנחה:

מנחה פנימי: ד"ר בני סלומון

השנה: תשפ"ב

תוכן עניינים

4	1 תקציר	1
5	1.1 מבוא	1.1
5	1.2 הגדרת הבעיה	1.2
5	1.3 שיטות עבודה וטכנולוגיות	1.3
6	2 למידת מכונה ולמידה عمוקה	2
6.....	2.1 למידת מכונה	2.1
6.....	2.1.1 בעית הסיווג	2.1.1
6.....	2.2 למידה عمוקה	2.2
7	2.2.1 רשות עצבית מלאכותית	2.2.1
8.....	3 מודלי למידה	3
13.....	4 תהליכי ביצוע הממחקר	4
13.....	4.1 חקר מקדים	4.1
13.....	4.2 עיבוד נתונים	4.2
14.....	4.3 בניית מודלים	4.3
15.....	4.4 אופן ניתוח תוצאות המודלים ושיפורם	4.4
15.....	5 תוצאות	5
15.....		QDA 5.1
25.....		LDA 5.2
37.....		GNB 5.3
52.....		CNN 5.4
52.....	מודל CNN שכבה 1, 32 פילטרים	5.4.1
64.....	מודל CNN שכבה 1, 64 פילטרים	5.4.2
76.....	מודל CNN 2 שכבות, 32 פילטרים	5.4.3
88.....	מודל CNN 2 שכבות, 64 פילטרים	5.4.4
100.....	מודל CNN מורכב	5.4.5
113.....	6 מסקנות	6
113.....	6.1 מסקנות כלליות עבור מודלים פרימיטיביים	6.1
113.....		LDA 6.1.1
113.....		QDA 6.1.2
113.....		GNB 6.1.3
114.....	6.2 מסקנות עבור מודלי CNN	6.2
114.....	6.2.1 מסקנות כלליות	6.2.1
115.....	6.2.2 מסקנות ביחס למודלים	6.2.2
116.....	6.2.3 השוואה ביחס למחקר	6.2.3

117.....	6.3 משפחות בעלות דמיון.....
117.....	6.3.1 משפחות בעלות דמיון – כללי
118.....	6.3.2 משפחות בעלות דמיון – מודלים ספציפיים.....
119.....	6.4 זמני ריצה
120.....	6.5 סיכום
120.....	7 מקורות

1 תקציר

בועלמנו כיים עולה השימוש בטכנולוגיה בצורה כבירה ועם עלייה זו, צדים אין ספור איזומים על המשתמשים השונים. מתקפות הסייבר בעולם מהוות סיכון מרכזי לפגיעה במרחב הקיברנטי של היעד במטרה לאגונוב ממנו מידע ואך להסביר לו נזק.

בנוסף לכך, בתחום טכנולוגי אחר אשר מתפתח במהלך השנים האחרונות – מצל שבו מנסים לדמות את יכולות החשיבה האנושיות ומבנה מוח האדם באמצעות אמצעים טכנולוגיים. וזאת זה מסתעף לתחומיים רבים, אם כי הנושא העיקרי שמעניין אותנו הוא "הלמידה העמוקה" – תחום הנגורן מתוך מידת המכונה, אשר מתרבס על רשותות ניוירונים (מודל מתמטי חישובי שפותח בהשראת תהליכי מוחיים/קוגניטיביים) בעלות מספר רב של שכבות. פרויקט זה בא לשלב בין שני העולמות הללו. מטרתנו היא לדמות, לשחרר ו אף להשוו את תוכאותינו לניסוי [1] שנערך באוניברסיטת סאן חוזה חלק מתחום שבוצעה בתחום. הרעיון הכללי הוא לבנות מודל למידה עמוקה אשר בהינתן תמונה של נזקה (Malware) יידע לסוגה למשפחת נזקות ספציפית.

תמונה של נזקה – זהה בעצם תוכחת המרה של נזקה לכדי תמונה. כך נוכל לבצע סיוג על בסיס למידה עמוקה באמצעות עיבוד וניתוח התמונה.

בנוסף לכך, נרצה להשוו את תוכאות הסיוג כתוצאה שימוש ניורוני קומבולוציונית (CNN) לתוכאות סיוג שונות, המבוססות על מודלים שונים מעולמות למדת המכונה, אותם נבנה. זאת כדי לנסות ולבדוק עלות אל מול תועלת (כגון: זמן ריצה, ביצועים, דיקן המודלים וכו') בין מודלים פרימיטיביים אל מול מודלים חדשניים כדוגמת CNN. מכיוון שמדובר במודל סיוג נתונים, התוצאות אותן נרצה לספק יהיו הסתברויות שנעו בין 0 ל-1, כך שכך שנתקרב ל-1 אזי מדובר בזיהוי נכון.

בפרויקט זה בנוינו ארבעה מודלים. שלושה מהם, הם מודלים פרימיטיביים שנטרכם להוות נקודת ייחוס למודל ה-CNN אותו רצינו לחקור, והם: LDA, QDA, GNB. כל אחד מלאה רץ (אימון ומבחן) על מאגר הנתונים כאשר כל מודל נחקר בשלושה אופנים שונים, כך שבעל אחד מהם התמונות במאגר הנתונים הוגדרו להיות: 32x32, 64x64, 128x128 פיקסלים.

הישגי הפרויקט אכן מוכחים את עליונות השימוש במודלים מסוג CNN אל מול המודלים השונים שנבדקו בפרויקט. בנוסף, ביחס למחקר עליו ביססנו את הפרויקט, ניתן לראות כי הצלחנו לשפר את אחוז הדיוק בכלל מודלי-H-CNN תוך התגברות על קשיים בזיהוי משפחות אשר מופיעים במחקר.

למשל, עבור *2 Convolutional Layers, 64 Filters 128 × 128 CNN* המחבר במאמר השיג כ-95.7% הצלחה ואילו מנגד, הצלחנו להשיג כ-99.7% אחוז הצלחה.

1.1 מבוא

תשתיות החברה כוים מתבססת ברובה על מחשבים אוינטראנט. אבטחת הסייבר היא אומנות ההגנה על רשתות, מכשירים ונוטונים מפני גישה בלתי מורשת או שימוש פלילי ובנוסף מהוות אבן יסוד בכל הנוגע להבטחת הסודיות, תקינות ואמינות המידע. לפי המחבר [1] עליו אנחנו מתבוססים, בשנת 2018 הובחנו כ-669 מיליון סוג נזקים חדשים. תחום העולה גם הוא בעולם הטכנולוגי בעקבות תוכנות תוכאות וממצאים מחקרים טובים הוא תחום הבינה המלאכותית. בתחום זה נוצרת הלמידה העמוקה אשר בה נעסק בפרויקט זה. למידה عمוקה היא תחום העוסק באלגוריתמים ומודלים בהשראת מבנה ותפקיד המוח, מודלים אלו נקראים רשות עצביות מלאכותיות. ניתן לראות שילובים בין העולמות הללו בתוכנות אנטוירוס שונות אשר מבוססות את יכולת הבדיקה שלhn על בסיס בינה מלאכותית על מנת לאתר ולזיהות פעולות ותהליכי זדוניים.

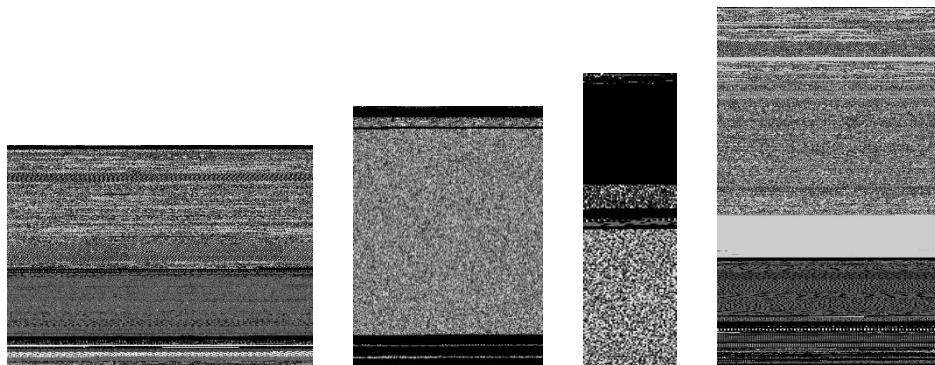
על קב העובדה כי תהליכי וকבצים זדוניים משתמשים בקצב מהיר, קיימים אתגר עבור עולם אבטחת הסייבר המתבסס על בינה מלאכותית לזהות ולנטר את אוטם האויומים. בנוסף, מודלים ואלגוריתמים מבוססי למידה عمוקה מצרכים יכולות מחשב גדולות על מנת לספק תוצאות בזמן סביר.

1.2 הגדרת הבעיה

הבעיה אותה הפוך בא לפתור היא בעיית סיוג נזקים למשפחות באמצעות למידת מכונה. קלומר, סיוג שיטתי של אויומים אלו, המאפשרים על מערכות תקשורת אוינטראנט המהוות תשתיות עבור ארגונים רבים. ביצוע והצלחה בפתרת בעיה זו, הכרך בפתרונות רבים והתאמות מיוחדות.

1.3 שיטות עבודה וטכנולוגיות

- 1.3.1 הפוך וקוד המקורי מונוהים באמצעות GitHub.
- 1.3.2 סביבת העבודה שבה השתמשנו היא Google Colab אשר תורץ על .Google Notebook
- 1.3.3 הפוך יכול לנכתב ב-**Python**, כאשר השתמשנו במגוון ספריות שונות, כגון:
 - אשר שימשה לייצרת מודל CNN-Keras.
 - אשר שימשה לייצרת המודלים Scikit-Learn, QDA, GNB, LDA.
 - אשר סייעו בעיבוד המידע Pandas & Numpy.
- 1.3.4 מאגר הנתונים שעליו התבוסטו הוא Malimg Dataset המכיל 9764 תמונות מ-25 משפחות נזקים שונות בסקללה של גווני הצבע האפור. את המאגר נמייר לשילושה גודלים שונים (פיקסלים) והם: 32x32, 64x64, 128x128. זאת על ידי שימוש בספריית Pillow.



איור 1 : דוגמאות לתמונות של נזקים ממאגר המידע

באיר (1), ניתן לראות דוגמאות לתמונות אשר נמצאות במאגר המידע. תמונות אלו מייצגות נזקים שהומרו לכדי תמונות כאשר תחילך המרה בוצע על-ידי המרה של כל Byte בקובץ ה-Exe למספר בין 0 ל-255. לבסוף נקבל מערכת חד מימדי של פיקסלים קר שמה שנותר לעשות זה להפוך אותו למערך דו-ממדי שהוא את התמונה.

2. למידת מכונה ולמידה عمוקה

2.1. למידת מכונה

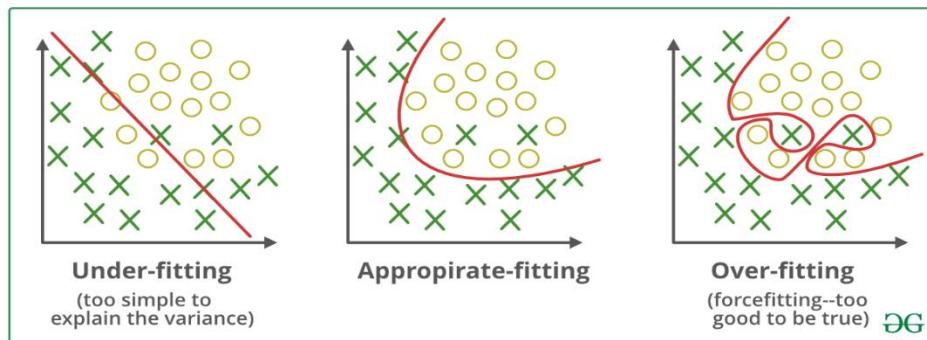
הת-הomonיות במדעי המחשב ובבינה מלאכותית. התחום עוסק בפיתוח אלגוריתמים המאפשרים למחשב ללמידה מותר דוגמאות, ופועל במגוון שימושים חישוביים בהן התקנות הקלאסי איננו אפשרי.

תחומי זה מכיל שלושה סוגים למידה עיקריים והם:

- **למידה מונחית (Supervised Learning)**: מאופיינת על ידי תיאוג מאגר המידע לקטגוריות, כרך שบทהlixir הלמידה של המערכת היא תוכל להשוות את תוצאותיה לתיאוג שהוגדר מראש, ובעצם ללמידה לסוג את המידע על סמך תשובות שהוגדרו, תהlixir זה יקרא שלב האימון (Train). לבסוף, המערכת תידרש לסוג קליטים חדשים שאיננה מכירה לקבוצה המתאימה, תהlixir זה יקרא שלב הבדיקה (Test).
- **למידה בלתי מונחית (Unsupervised Learning)**: מאופיינת על ידי יכולת לקבל מידע שאינו מתואם לקטגוריות, ובעצם לדעת לנתח ולאגד אותו ללא הטעבות אדם.
- **למידת חיזוק (Reinforcement Learning)**: מאופיינת על ידי יכולת להעניק תגמול על התנהגות נכונה ו- "להעניש" כאשר התנהגות אינה נכונה. המערכת מקבל משוב חלקי בסיסים העבודה ותדע להסיק מסקנות לגבי השלבים אשר אותם יש לשפר/לשמר.

פרויקט זה מתמקד בעיקר בלמידה מונחית, כאשר תיגנו את התמונה ל-25 משפחות שונות של נזקוקות.

בעת אימון מודלי למידת מכונה, יש להיזהר מ מצב של התאמת יתר (Overfitting), כמתואר באירור (2), אשר עלול לגרום לכך שהמערכת תדע לסוג ביצורה טוביה את הדאטה שניתן לאימון, אך לא תצליח לעשות זאת עבור מידע חדש.



איור 2: דוגמאות לסוגי Fitting

בנוספ, עוד בעיה העולה עלולה להיות היא "Curse of Dimensionality". לפיה, מספר הדוגמאות נדרש על מנת שתהיה יכולה להעריך פונקציה שרירותית מסוימת בדיקת גובה, גדול בצורה אקספוננציאלית ביחס למספר הקליטים אשר אותה פונקציה מקבלת.

2.1.1. בעיית הסיווג

בעיית הסיווג היא בעיה הנוגרת מתחומי הלמידה המונחית (Supervised Learning) אשר לפיה, בהינתן קלט אל המודול נרצה להחליט לאיזו מחלקה הוא שייך.

2.2. למידה عمוקה

היא חלק ממשפחה רחבת יותר של שיטות למידת מכונה המבוססות על רשות עצביות מלאכותיות בשילוב עם למידת מאפיינים. הרעיון מאחורי שיטה זו היא יכולת למד מחשבים לבצע עבודה אשר נראה טבעי למוח האדם אך מסובכת לביצוע המחשב.

בלמידה عمוקה המודל יקבע אילו מהפיצ'רים רלוונטיים להסקת המסקנות ואילו לא, לעומת זאת למידת מכונה, אשר בה יש לבצע עבודת הכנה מוקדמת של סידור הדטה ובחירה הפיצ'רים המתאימים.

במקרה שלנו, הפיצ'רים באמצעות נרצה לפתור את בעיית הסיווג הם הפיקסלים של התמונות אותן נכניס קלט למודלים.

2.2.1 רשת עצבית מלאכותית

רשת עצבית מלאכותית (Artificial Neural Network – ANN) היא מודל מתמטי חישובי, שפותח בהשראת תהליכי מוחיים או קוגניטיביים המתרחשים ברשת עצבית טبيعית ומשמש בסוגרת למידת המכונה. הרשת מכילה לרוב מספר גדול של יחידות מידע המקשרות זו לזו, כאשר צורת הקישור בין היחידות מצירפת את אופן חיבור הנירונים במוח ומכאן נובע שמה.

ברשת מספר רב של נירונים כאשר כל אחד מהם מבצע חישוב פשוט ויחסית. אל הנירון מתתקבל קלט אשר עליו מבוצע עיבוד באמצעות נוסחה מוגדרת שתוצאה מהו הפלט של אותו הנירון. פלט זה מועבר להלאה אל נירונים אחרים ברשת אשר מבצעים תהליכי דומה.

רשת מאופיינית על ידי מספר תכונות כגון, צורת חיבור הנירונים המהווה את האופן בו הנירונים מחוברים האחד לשני בין השכבות השונות, פונקציית הפעלה (Activation Function) הקובעת האם קלט הנירון ברשת הוא חשוב או לא ואת באמצעות פעולות מתמטיות, ומשקל נירונים אשר קבועים עבור כל נירון את חשיבותו ביחס למודל עצמו ולהמשך התהליך.

רשת הנירונים מורכבת משלוש שכבות עיקריות (איור 3) – שכבת הקלט, שכבת העיבוד ושכבת הפלט.

- **שכבת הקלט (Input Layer)**: שכבה זו מקבלת את המידע הראשוני אשר אמרור להיקלט אל תוך המודל. אל כל נירון השير לשכבה זו מתתקבל ערך מידע יחיד. משכבה זו עובר המידע אל השכבה הבאה בתווך – השכבה החבוייה (Hidden Layer).

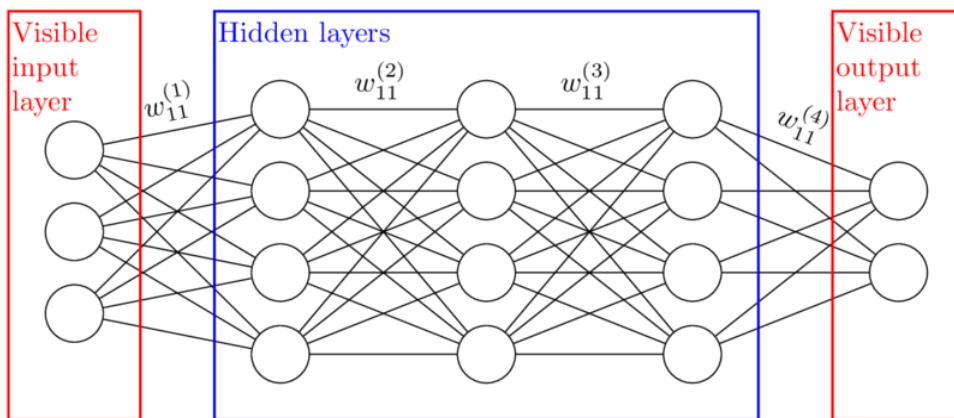
במקרה שלנו, שכבת הקלט היא הפיקסלים של התמונה, לפיכך עברו מודל של $A \times A$ מספר הנירונים בשכבת הכניסה תהיה A^2 . בנוסף, כל פיקסל יהווה מאפיין (Feature) עבור המודל.

- **שכבה חבוייה (Hidden Layer)**: שכבה זו ממוקמת בין שכבת הקלט ושכבת הפלט. מטרת השכבה היא ביצוע חישובים וטרנספורמציות לא לינאריות על הנתונים אשר נקלטו בה. לבסוף, שכבה זו תעביר את תוצאותיה אל שכבת הפלט (Output layer).

במקרה שלנו, השכבה החבוייה תורכב משכבות קונבולוציה שונות (אשר יפורטו בהמשך) כדי לבצע חישובים מתאימים.

- **שכבת הפלט (Output Layer)**: השכבה הסופית ברשת אליה מתקבלים כלל התוצאות והتوزעות מהמודל.

במקרה שלנו, תוצאות המודל אשר יתקבלו בשכבה זו יהיו מספרים בין אין אפס לאחד, אשר יהוו הסתברות של הקלט להיות שיר למשפחה מסוימת. שכבת הפלט שלנו מורכבת מ-25 נירונים אשר כל אחד מהם משפחת נזקوت שונה.



אייר 3: מבנה רשת נירונית

3 מודלי למידה

מודל למידת מכונה הינו אלגוריתם אשר מטרתו היא פתרון בעיה כלשהי. במקרה שלנו, בעיה זו היא בעיית הסיווג. קיימים מספר אלגוריתמים ושיטות פתרון. שיטות אלה בכללותן מנוסות, בעזרת כלים מתמטיים וסטטיסטיים, למצוא דפוסי התנהגות בתופעות כלשהן אשר יסייעו בפתרון הבעיה. את תופעות וביעות אלה ניתן לתאר במגוון צורות, אך כאמור רוצים להמיר אותן לכדי מודל מתמטי אותו ניתן לנסות לפתור בשיטות הידועות, יש לתאר אותן באמצעות מונחים. מונחים אלו יכולים להיות:

- מאפייני התופעה (Features) אשר בחלקם יכולים להיות פרמטרים כמוותים ובחלקם الآخر, פרמטרים שהינם יותר אבסטרקטיים אשר צריך לקבוע כיצד ובאיזה אופן יש להמירם לכדי משתנים כמוותים. במקרה מהמרקם, אנו קובעים אלו מהמאפיינים רלוונטיים עבור המודל ולפיכך גם מספקים לו את מאפייניהם אלו.
- במקרים אחרים, נספק למודל את כל המאפיינים ונitin לו לקבוע באותו אילו מהמאפיינים רלוונטיים עבורו.
- פلت המודל, אשר מהווה את פתרון הבעיה. במקרים מסוימים פתרון הבעיה יכול להיות משתנה בדיד אשר יכול לייצג הסתברות כלשהי, ובקרים אחרים מספר משתנים אשר כל אחד מהם מאפיין תוצאה אפשרית אחרת שיכולה להתבלבב ביחס לקלט הנתון. קיימים מגוון רחב של אפשרויות נוספות, כאשר כל אפשרות נקבעת בהתאם לסוג הבעיה.

3.1 (Quadratic Discriminant Analysis) QDA

זה מודל/אלגוריתם למידת מכונה העוסק בעיית הסיווג. כולם, מטרתו לסוווג בין 2 מחלקות (Classes) או יותר, באופן כזה שעבור כל מחלוקת, לבסוף, מותאם גאומטרי אשר תוחם בתוכו את כלל הערכים האפשריים עבור מחלוקת זו כתואר באירור (4).

מודל זה עובד על ידי חישוב מאפיינים סטטיסטיים על הנתונים המסופקים לו. עבור משתנה קלט בוודוד יוגדרו הממוצע והשונות שלו עבור כל מחלוקת (Class). כאשר מדובר משתני כניסה רבים, קיים ייצוג מטריציוני באופן דומה.

תוצאות חישוב סטטיסטי זה משמשים לבסוף בכך לייצג את המודל המדובר ולבסוף גם יישמו לביצוע תוצאות בהמשך.

הנתונים יסוד אשר מודל זה מניח על המידע המתתקבל אליו:

- הנתונים מתפלגים נורמלית.
- לכל מחלוקת יש שונות משותפת (Covariance) אחרת אשר מאפיינת אותה.

נגידר את חישוב הערך הממוצע (\bar{x}_M) לכל קלט (x) עבור כל מחלוקת (C) באופן הבא:

$$\mu_{MLC} = \frac{1}{N} \sum_{n=1}^N X_n$$

נגדיר שונות σ_{ML}^2 למחלקה C באופן הבא:

$$\sigma_{MLC}^2 = \frac{1}{N} \sum_{n=1}^N (x_n - \mu_{ML})^2$$

כאשר N זהו מספר המופעים של אותה מחלקה.

(Linear Discriminant Analysis) LDA 3.2

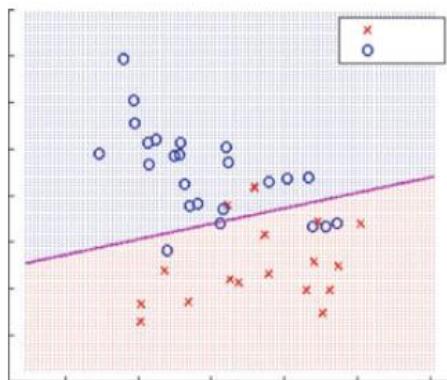
מודל זה מתבסס ברובו על מודל ה-QDA, אם כי השוני העיקרי ביניהם הוא שבניגוד ל-QDA, LDA משתמש בשונות משותפת זהה עבור כל המחלקות הקיימות.

הנחה יסוד אשר מודל זה מניח על המידע (המידע המתkeletal למודל):

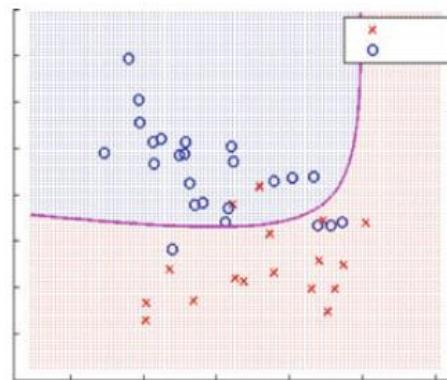
- הנתונים מתפלגים נורמלית.
- לכל מחלקה יש את אותה השונות המשותפת.

יתר החישובים מתבצעים באופן דומה לחישובי QDA מלבד ההנחהות המצוינות לעיל.

LDA



QDA



אייר 3 : LDA Vs QDA

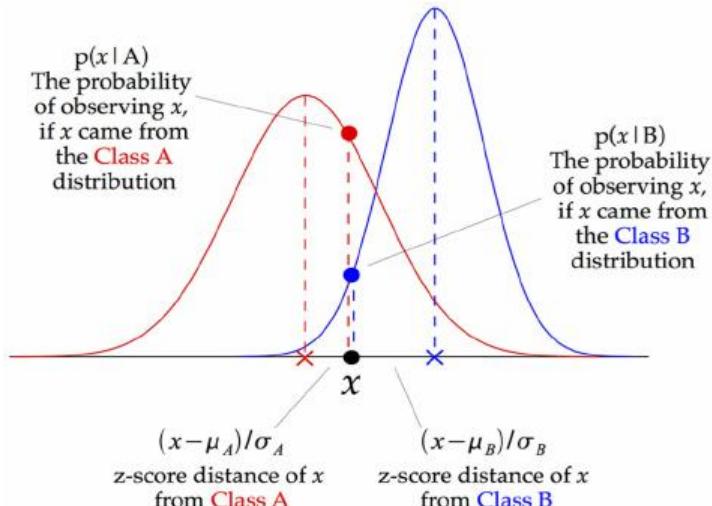
(Gaussian Naive Bayes) GNB 3.3

גם מודל זה מתבסס ברובו על מודל ה-QDA. השוני העיקרי בין שני המודלים הוא שמודל ה-GNB מניח כי מטריצות השונות המשותפות המתkeletalות הן מטריצות אלכסוניות.

הנחה יסוד אשר מודל זה מניח על המידע (המידע המתkeletal למודל):

- הנתונים מתפלגים נורמלית.
- לכל מחלקה יש מטריצה שונות משותפת אחרת אשר מאפיינת אותה, כאשר מטריצה זו היא מטריצה אלכסונית.

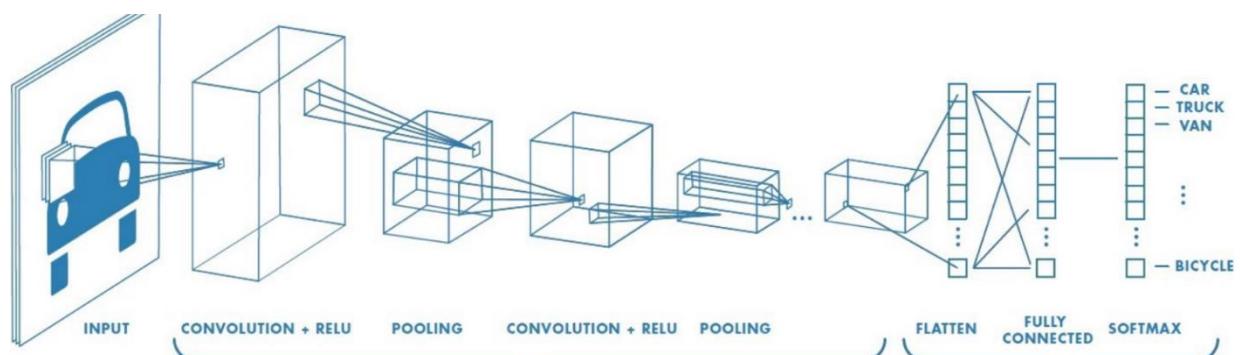
- אין תלות סטטיסטית בין ערכי האלכסון (בין הקלאסים השונים).
- יתר החישובים מתבצעים באופן דומה לחישובי QDA מלבד ההנחהות המצוינות לעיל.



אייר 4.

(Convolutional Neural Network) CNN 3.4

זהי סוג של רשת נוירונים המשמשת בעיקר לניטוח תכונות. הרשת מבוססת על שכבות כניסה (Convolution) הממומשות על ידי העברת פילטר על גבי הקלט ומשתמשת בפעולת הקונבולוציה (פחות בשכבה אחת). ברשת CNN, בדרך כלל נתיחס לצבעי התמונה המשמשת כקלט כמערך תלת ממדי כאשר כל מערך בו מאופיין על ידי אחד משלושת הצבעים: אדום, ירוק וכחול (RGB). בנוסף, מקדמי הקונבולוציה נלמדים עצמאית במהלך אימון המודול. כאשר משתמשים ב-CNN על תמונות, נקבל עמידות רבה (Invariance) ביחס לתרנספורמציות על נתונים אשר תעזר להקטין את הסיכויים לקבלת overfitting אשר עלולים להגרם עם ריצת המודול.



אייר 5: תהליכי רשת קונבולוציה

3.4.1 קונבולוציה

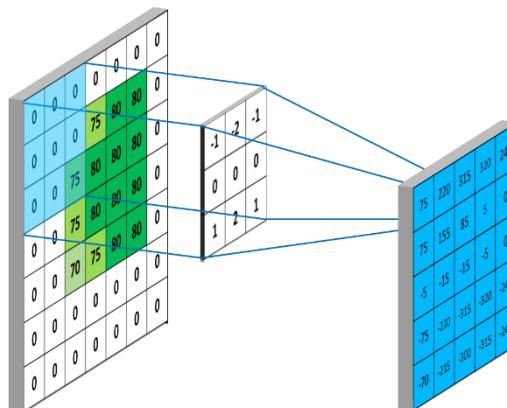
במקרה שלנו נשתמש בקונבולוציה בדידה אשר מהווה פעולה מתמטית של סכימת מכפלות האיברים בין שתי מטריצות כמתואר באיור 7).

$$\begin{array}{|c|c|c|} \hline 1 & 2 & 3 \\ \hline 2 & 0 & 0 \\ \hline 7 & 9 & 1 \\ \hline \end{array} * \begin{array}{|c|c|c|} \hline 3 & 2 & 0 \\ \hline 3 & 0 & 1 \\ \hline 0 & 5 & 2 \\ \hline \end{array} = \begin{array}{|c|c|c|} \hline 1*3=3 & 2*2=4 & 3*0=0 \\ \hline 2*3=6 & 0*0=0 & 0*1=0 \\ \hline 7*0=0 & 9*5=45 & 1*2=2 \\ \hline \end{array}$$

אייר 6: קונבולוציה

3.4.2 פילטר (Filter)

חלון מטריציוני בגודל מסוים המכיל תבנית שבזרתת מתבצעת פעולת הקונולוציה על הקלט, כאשר המטרה הסופית היא למצוא דמיון בין התבנית לבין אזורים שונים על גבי התמונה. למשל, קווים ישרים, עיקולים, צורות, דפוסים וכו'. ככל שהרשות יותר عمוקה, כך המודל ידע לזהות אובייקטים ודפוסים יותר מורכבים.



איור 7: Filter

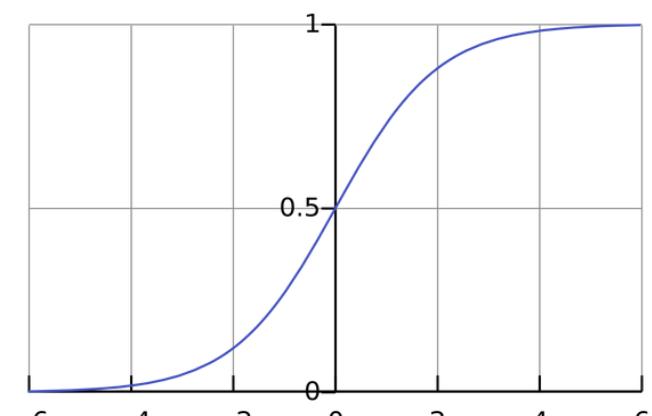
3.4.3 Activation Function

פונקציית אקטיבציה היא פונקציה מתמטית אשר עשויה להיות שונה בכל שכבה ושבה ברשות. פונקציה זו קובעת עבור כל ניירון השירך לשכבה האם הינו פועל או לא. כלומר, קובעת אם הקלט של כל ניירון רלוונטי לתהליכי החיזוי (או רלוונטי להיות קלט של ניירון אחר).

3.4.3.1 Softmax

זהו סוג של פונקציית אקטיבציה אשר נהוג להשתמש בה בשכבה הפלט (Output Layer) של מודלי סיווג. פונקציה זו מקבלת וקטור תוצאות ומmirה אותו לכדי וקטור הסתברויות שלבסוף יהווה את פלט המודל.

$$\sigma(z)_i = \frac{e^{z_i}}{\sum_{j=1}^K e^{z_j}}, \quad \text{for } i = 1, \dots, K \text{ and } z = (z_1, \dots, z_k) \in \mathbb{R}^K$$

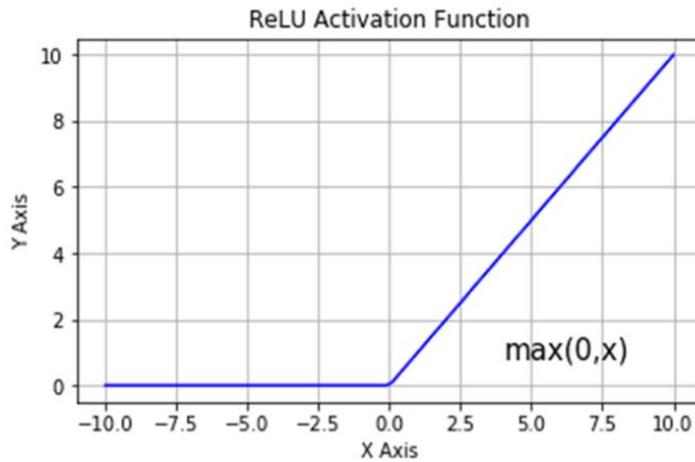


איור 8: פונקציית Softmax

ReLU 3.4.3.2

זהי סוג של פונקציית אקטיבציה אשר נהוג להשתמש בה כחלק משכבה הconvולוציה. פונקציה זו תעביר ארך וערך חיוביים ואילו ערכים שליליים תחליף ב-0.

$$f(x) = \begin{cases} x & \text{if } x \geq 0 \\ 0 & \text{if } x < 0 \end{cases}$$



איור 9: פונקציית ReLU

Feature Map 3.4.4

תוצר של שכבת הקונבולוציה אשר מכיל מיפוי מיוחד ביחס להיקן מאפיינים מסוימים יכולם להימצא בתמונה.

3.4.5 סוגי שכבות

Conv2D 3.4.5.1

שכבה זו מיישמת את פעולה הקונבולוציה על הקלט על ידי שימוש בפילטר הבניי ממטריצה (דו ממדית).

Max Pooling 3.4.5.2

שכבה זו מתבוננת על חלקיים, בגדים שהוגדרו מראש, ב-map Feature ולקחת מכל חלק את הערך המקסימלי המופיע בו. המטריה לסנן ערכיהם אשר עלולים להיות לא רלוונטיים להמשר התהיליך.

Dropout 3.4.5.3

טכниקה אשר באמצעותה נבחרים רנדומלית נוירונים אשר מהם יתعلמו במשך אימון המודל.

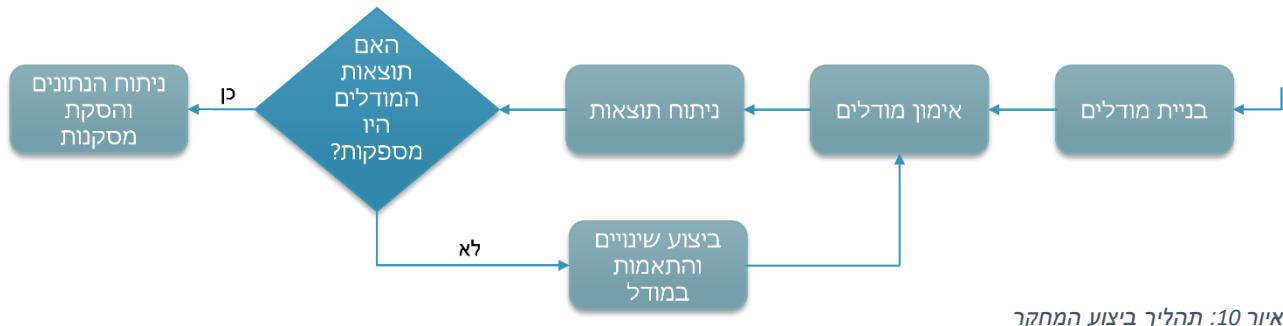
Dense 3.4.5.4

שכבה אשר בה כל נוירון מקבל קלט מיתר הנוירונים הנמצאים בשכבה הקודמת לה. שכבת זו משמשת לסייע התמונה בהתאם על הפלט שהתקבל משכבות הקונבולוציה.

Flatten 3.4.5.5

התהיליך שבו מטבחצת המטרה של המידע למערך חד-ממדי שיהווה קלט לשכבה הבאה בתור. בדרך כלל, נבצע פעולה זו לאחר שימוש בשכבת הקונבולוציה ולפני הכניסה לשכבת המוצא. זאת מכיוון ששכבת ה-Dense יודעת לקבל מערך חד-ממדי בלבד.

4 תהליך ביצוע המבחן



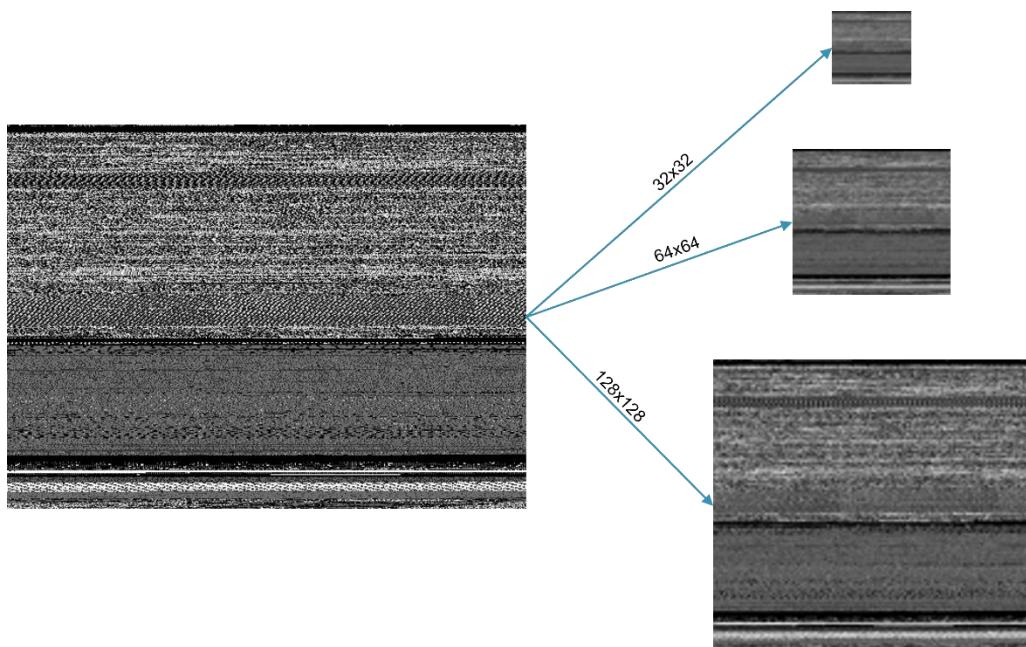
איור 10: תהליך ביצוע המבחן

4.1 חקר מקדים

תחילה, נדרש מאיתנו למידה של כל עולם למידת המכונה, הלמידה העומקה, הכלים והטכנולוגיות שאיתם נדרשנו להשתמש בפרויקט זה. לאחר מכן, עברנו לשלב חקר וניטוח המבחן המרכזי עלי התבססנו בפרויקט זה על מנת לבסס את הבעה שאיתה נרצה לפטור. לבסוף, שילבנו בין כל יכולות אלו לכדי עיצוב המבנה המרכזי ואפונ ביצוע הפרויקט.

4.2 שימוש נתונים

לפני תהליכי בניית המודלים, נדרשנו לעבד את מאגר המידע בו השתמשנו בכדי להתאיםו לבניית הפרויקט אותו הגדרנו בשלב החקר המקדים. בנוסף, יצרנו כלים אשר עזרנו לנו לייעל את תהליכי העבודה שלנו כוצאות ואת תהליכי העבודה על פלטפורמת הענן Google Colab. כפי שהוגדר במבנה הפרויקט נדרשנו לבצע שינוי רגולציה של מאגר התמונות לשולחה גדלים, והם: 32×32 , 64×64 ו- 128×128 . ככלומר, הפעולה אותה ביצענו הייתה לעبور על כל מאגר התמונות שלנו תוך שינוי גודלה המקורי של כל תמונה לכל אחד מהגדלים שצינו לעיל (תוך כדי פועל או בוצע מיפוי התמונות).



איור 11: הרמה לגודלים שונים

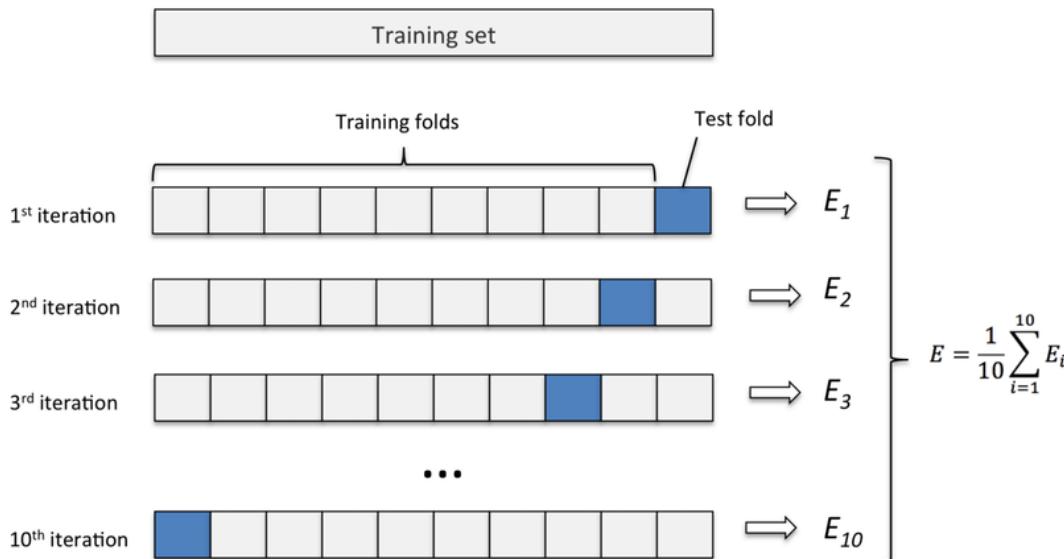
4.3

בנייה מודלים

עבור כלל המודלים, תחילת גמישור ונסדר את המידע הרלוונטי עליו יירצו המודלים. כאשר אנו מדברים על המודלים הפרימיטיביים, בהתחלה נשתח את מערכת התמונה וננרטל אותה. עבור מודלי CNN, נדרש רק לנरמל את מערכ התמונה (אין צורך לבצע שיטות מכיוון שמודול זה מסוגל לעבד עם מידע דו-ממד). בשלב הבא נרץ את המודול על ידי שימוש ב-K-Fold, אשר יdag לאופן חלוקת המידע באופן הבא, ראשית חלק זה ירץ כ-9 פעמים כאשר K, המיצג את אופן חלוקת המידע בין אימון ומחhn, ינוע בין 2 ל-10. כך שיחס התמונות שיבחרו לאימון יהיה $\frac{1}{K} - 1$ ויחס התמונות אשר יבחרו למחhn יהיה $\frac{1}{K}$, תוך כדי שמירה על יחס תקין בין כל המשפחות. חלק מסויל זה יבוצע תהליך החיזוי (סיווג) על המודול המאומן עבור קבוצת האימון והמחhn. לבסוף, על התוצאות שיתקבלו, יבוצע ניתוח והציג.

- **Stratified K-Fold:** על ידי שיטה זו נוכל ליחס חלוקת מידע מאוانت ולדאוג לכך שבעור כל חלקה אותה נגידר אשר תהיה תלולה ב-K, נוכל לבצע את תהליך החיזוי (סיווג) על כל המידע וכן נוכל להבטיח שלא ייווצר מצב שבו מידע מסוים יבדק רק עבור שלב האימון או שלב המבחן. תהליך זה מתרחש באופן הבא – עבור K נתון, יוגדרו K חלוקות (K_1, K_2, \dots, K_K) אשר כל אחת מהן תחולק כך שיחס התמונות שיבחרו לאימון יהיה $\frac{1}{K} - 1$ ויחס התמונות אשר יבחרו למחhn יהיה $\frac{1}{K}$. כל חלוקה שזו תגדיר חלוקה שונה של המידע. לכל חלוקה K_i , נבצע את תהליך החיזוי על סדרת האימון והמחhn. לבסוף, כל תוצאות תהליכי החיזוי K_i ימצאו ויחו את תוצאה ה-K. את תהליך זה נרץ כ-9 פעמים כאשר K ינוע בין 2 ל-10. המידע אותו אנו בחרנו להציג בפרויקט עבור כל מודל הוא בעצם ה-K אשר נתן את התוצאות הטובות ביותר בסדרת המבחן.

לדוגמה, ניתן לראות באIOR (9) את תהליך זה עבור $10 = K$. קר-Sh-E זוהי תוצאה החיזוי של איטרציה ספציפית עבור האימון והמחhn ו-E זו התוצאה הסופית עבור ריצה K כלשהי אשר קיימים לה שני רכיבים – תוצאה הסיווג לקבוצת האימון ותוצאה הסיווג לקבוצת המבחן.



AIOR 12 : K-Fold Validation

4.4

אופן ניתוח תוצאות המודלים ושיפורם
 כחלק מהתהlik התקדמות הפרויקט, פעלנו בשיטה מחזוריית של הבנת הקווים המנחים אותם נרצה לשמר
 ולעומת זאת הבנה של אלה אשר נרצה לשפר. ככלומר, כאשר אנו מתייחסים לשיטה זו עברו ניתוח תוצאות
 המודלים ושיפורם, תהlik זה כלל ניתוח תוצאות של אחריו נרצה להביא לכדי מיפוי הפרמטרים אשר
 משמשים את המודל. כל זאת בדגש על מודלי CNN אשר דרשו מאיינו מגוון רחב של ניסויים בהם שינינו
 את מבנה המודלים, שכבותיהם ואת הפרמטרים המסופקים להם. לאחר ביצוע ההתאמות חזרנו על תהlik
 זה מספר פעמיים נוספת עד אשר הגיענו לתוצאה רצiosa.

5 תוצאות

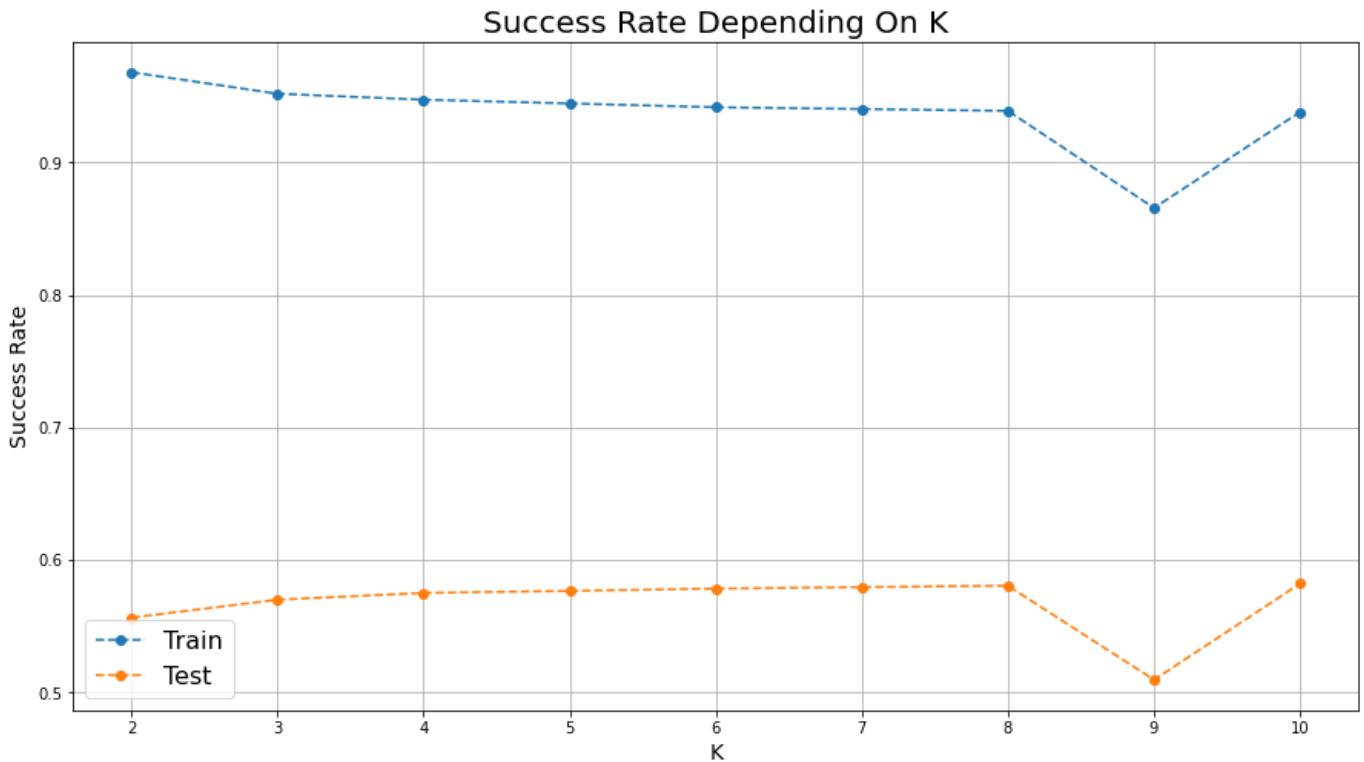
בכל אחד מהמקרים, נציג את תוצאות המודלים כאשר נבצע חלוקה למקרים ביחס לשכבות הקלט המאפיינת
 על ידי כמות הפיקסלים של כל תמונה במאגר התמונות הנבדק.

QDA 5.1

Size	Best K - Test	Test Accuracy	Best K - Train	Train Accuracy
32x32	10	58.224	2	96.815
64x64	4	54.915	ALL	100
128x128	7	48.976	ALL	100

טבלה 1: תוצאות QDA

32 × 32 5.1.1



Train (Includes test results)

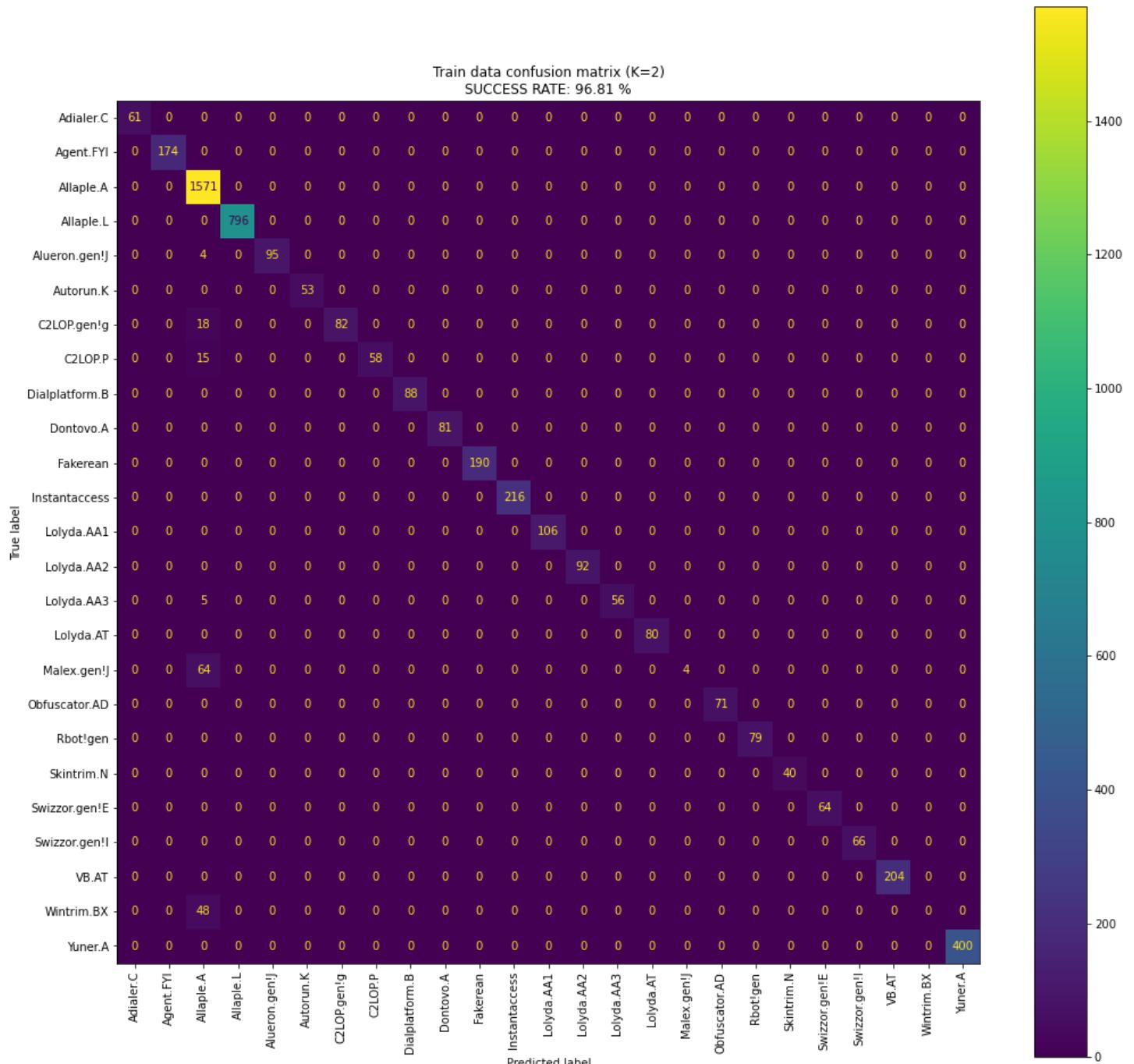
K = 2

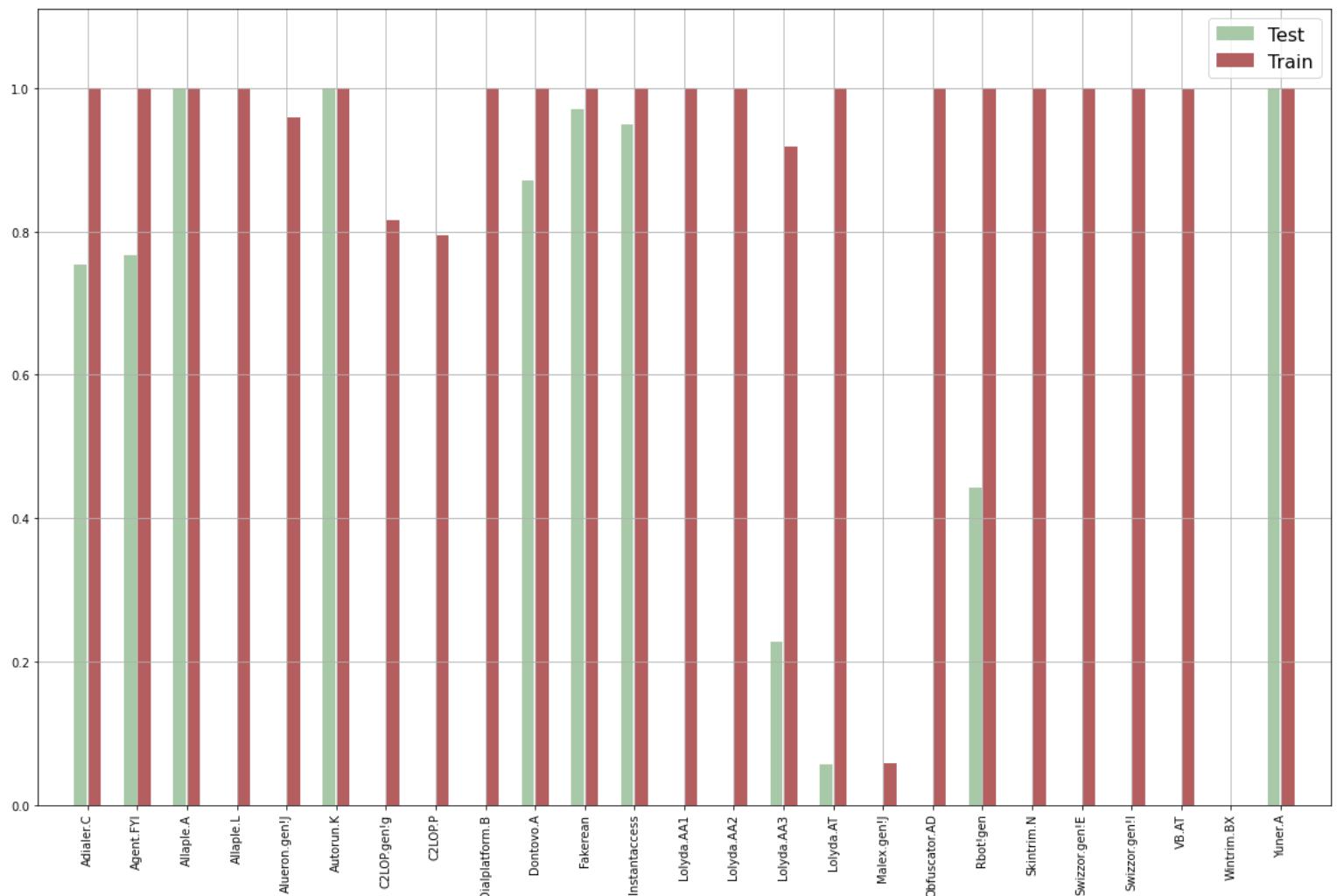
Success Rate (after normalizing):

- Train: 96.815 %
- Test: 55.653 %

Malware family with the highest false classification (after normalizing):

- Train: Wintrim.BX
- Test: Allaple.L





Test (Includes train results)

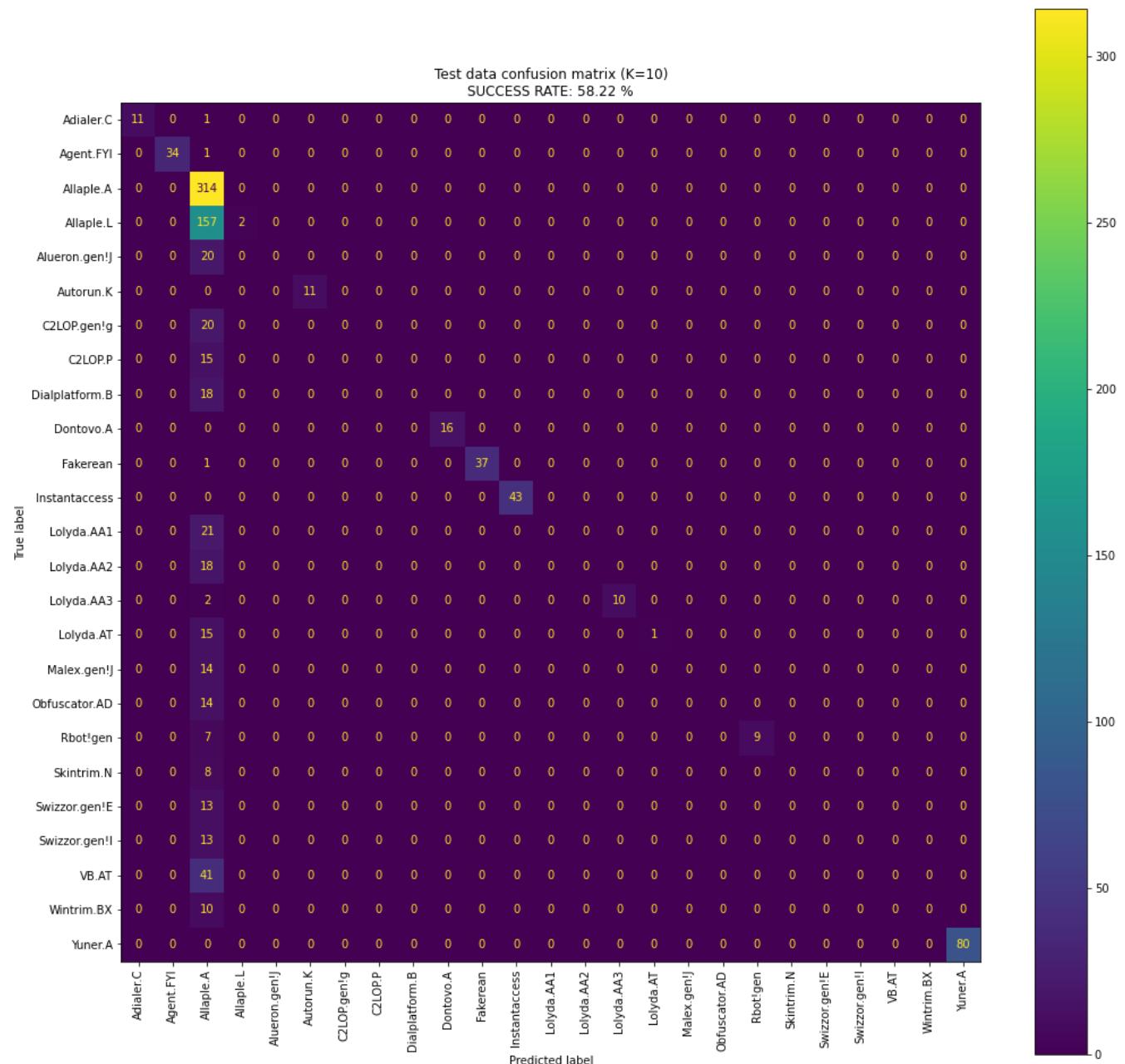
K = 10

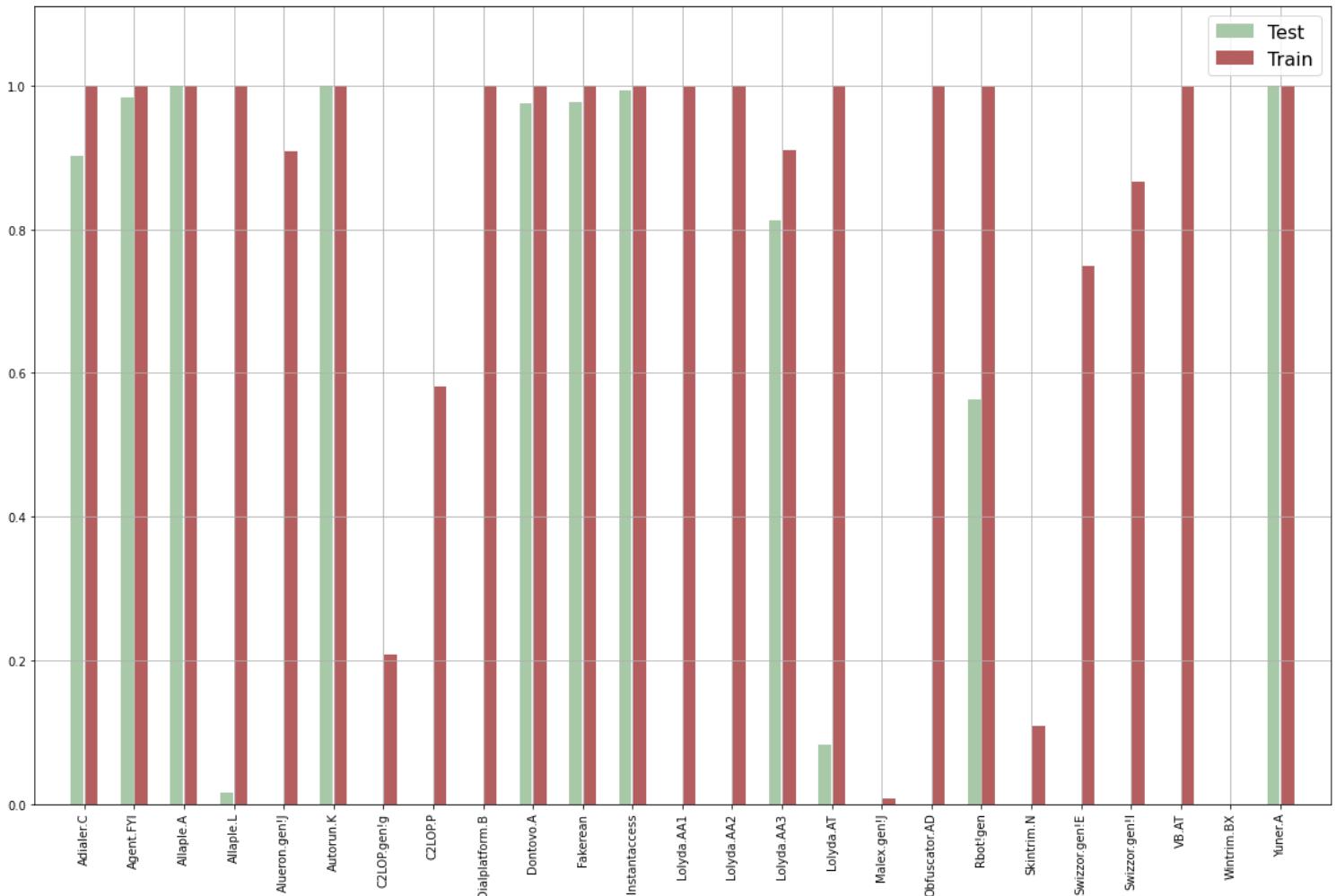
Success Rate (after normalizing):

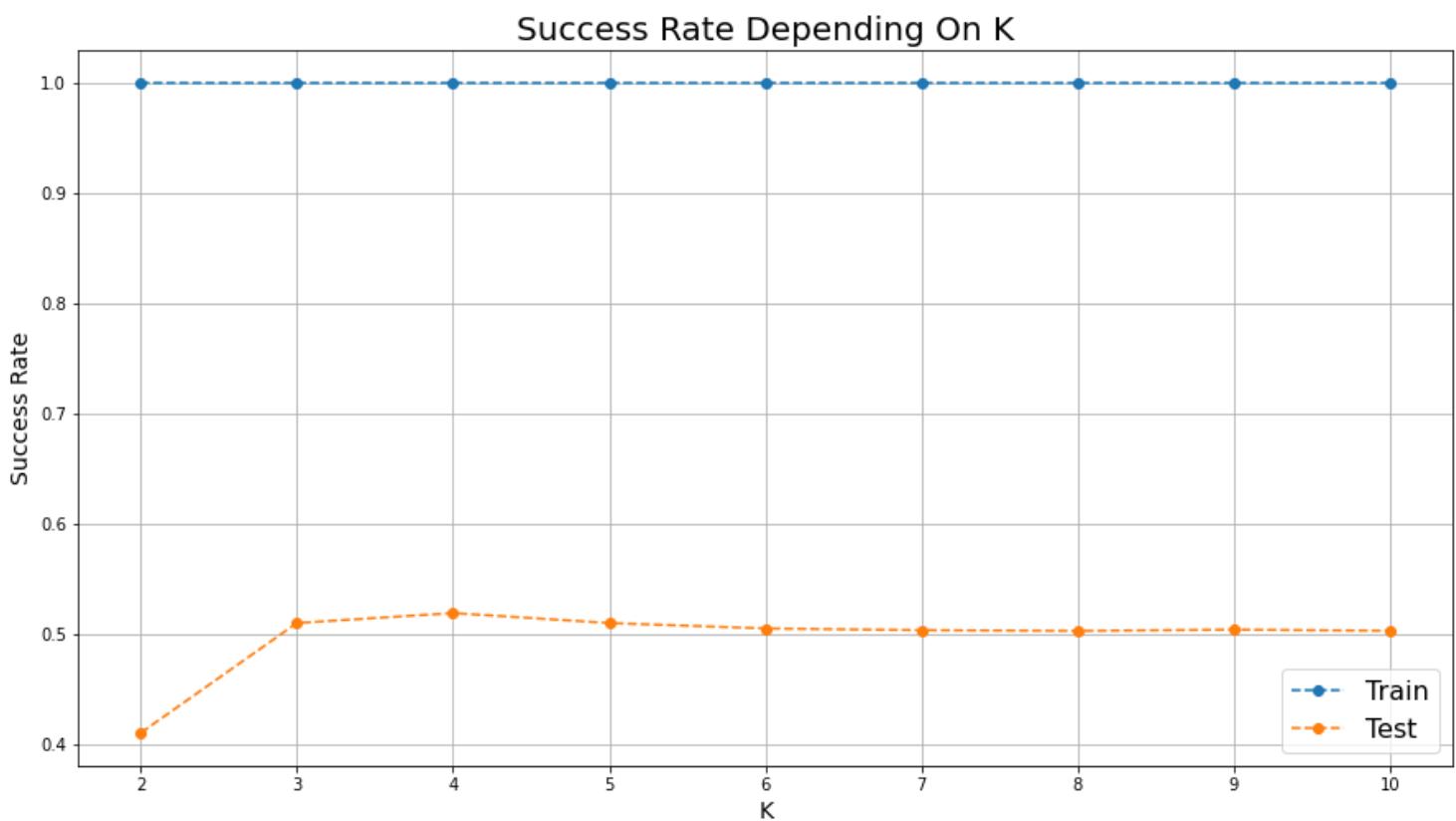
- Train: 93.821 %
- Test: 58.224 %

Malware family with the highest false classification (after normalizing):

- Train: Wintrim.BX
- Test: Alueron.gen!J







Train (Includes test results)

K = ALL

Success Rate (after normalizing):

- Train: 100.000 %

Malware family with the highest false classification (after normalizing): N/A

Test (Includes train results)

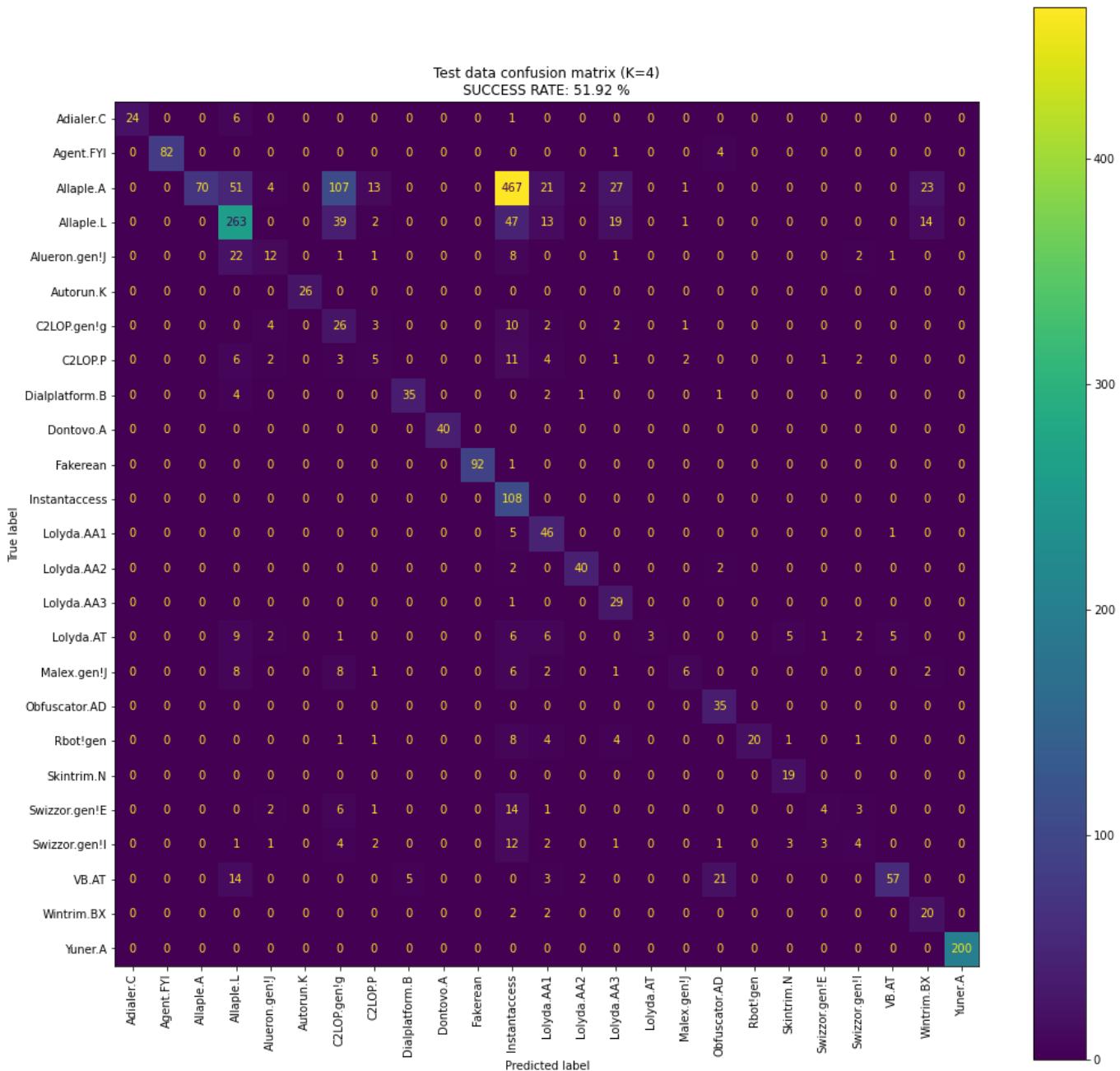
K = 4

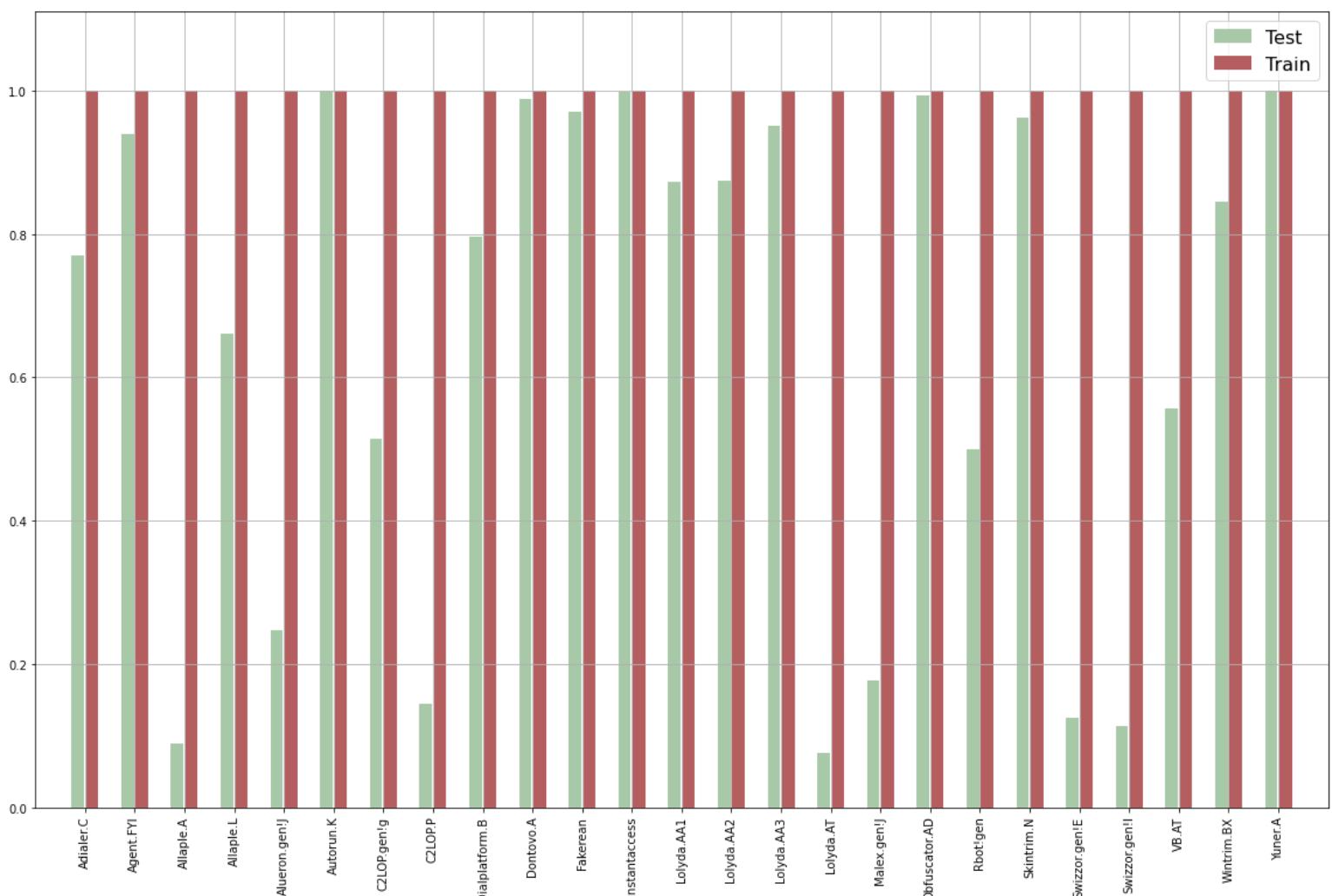
Success Rate (after normalizing):

- Train: 100.000 %
 - Test: 51.915 %

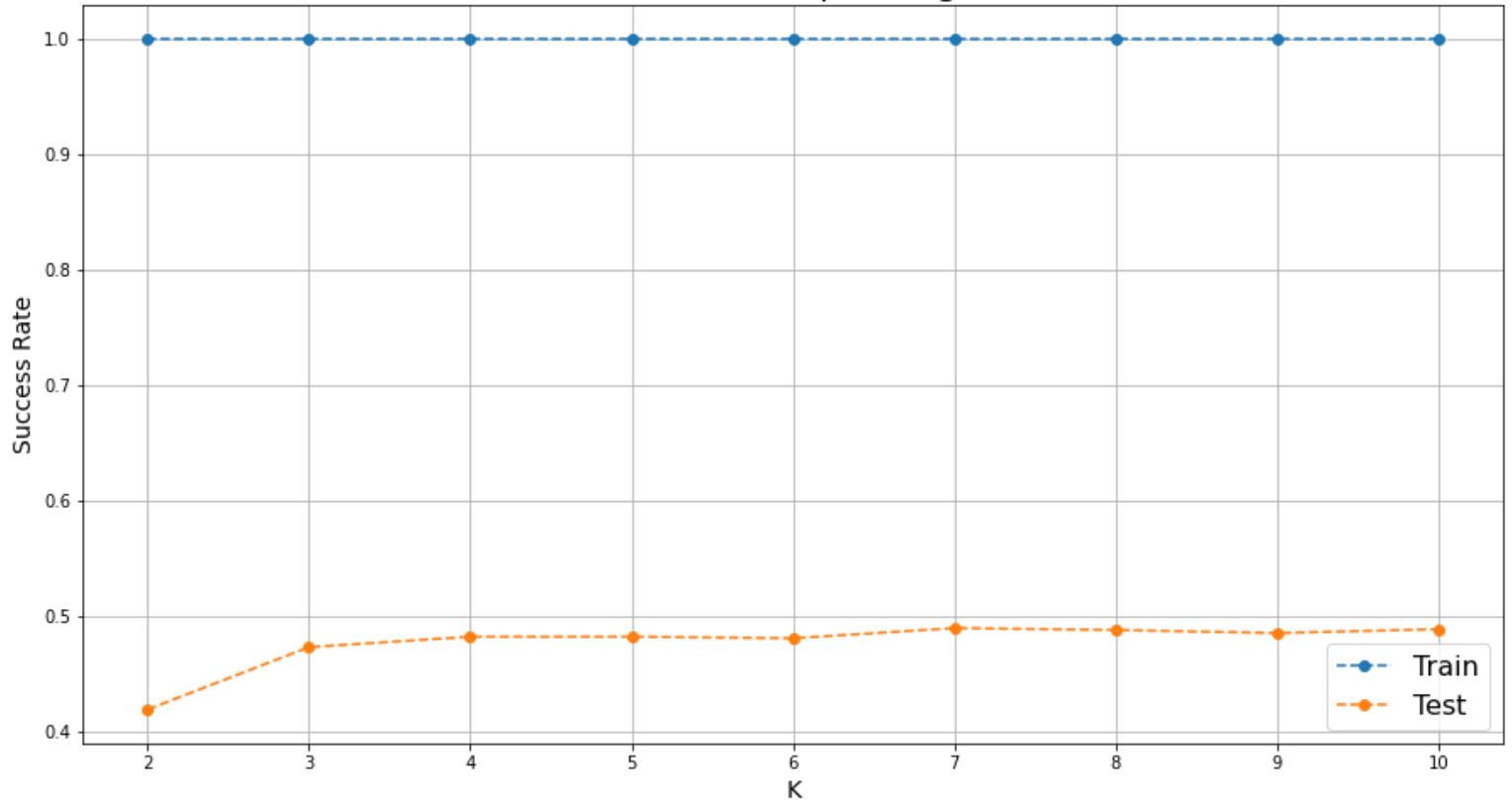
Malware family with the highest false classification (after normalizing):

- Train: N/A
 - Test: Lolyda.AT





Success Rate Depending On K



Train (Includes test results)

K = ALL

Success Rate (after normalizing):

- Train: 100.000 %

Malware family with the highest false classification (after normalizing): N/A

Test (Includes train results)

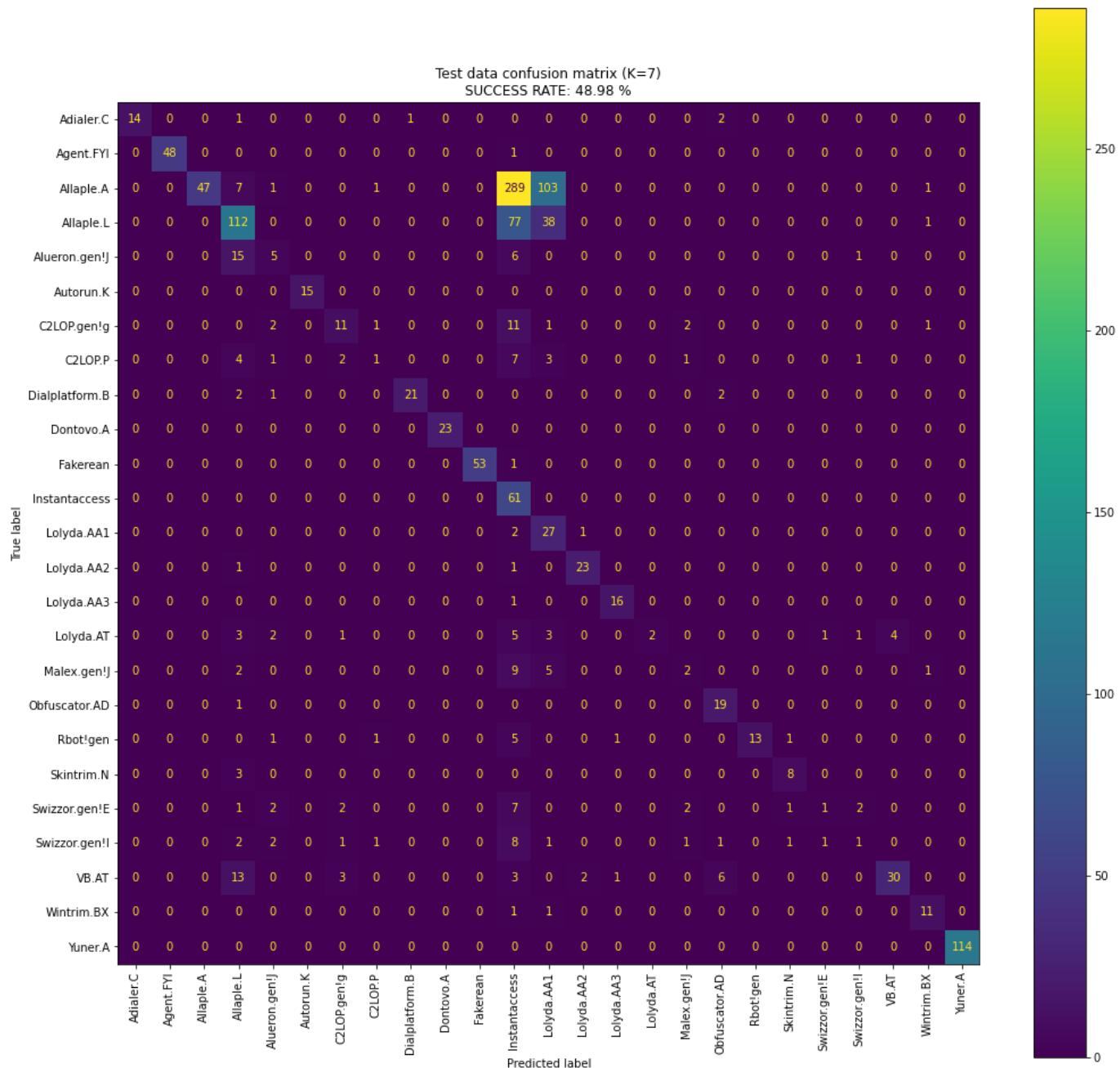
K = 7

Success Rate (after normalizing):

- Train: 100.000 %
- Test: 48.976 %

Malware family with the highest false classification (after normalizing):

- Train: N/A
- Test: Swizzor.gen!I



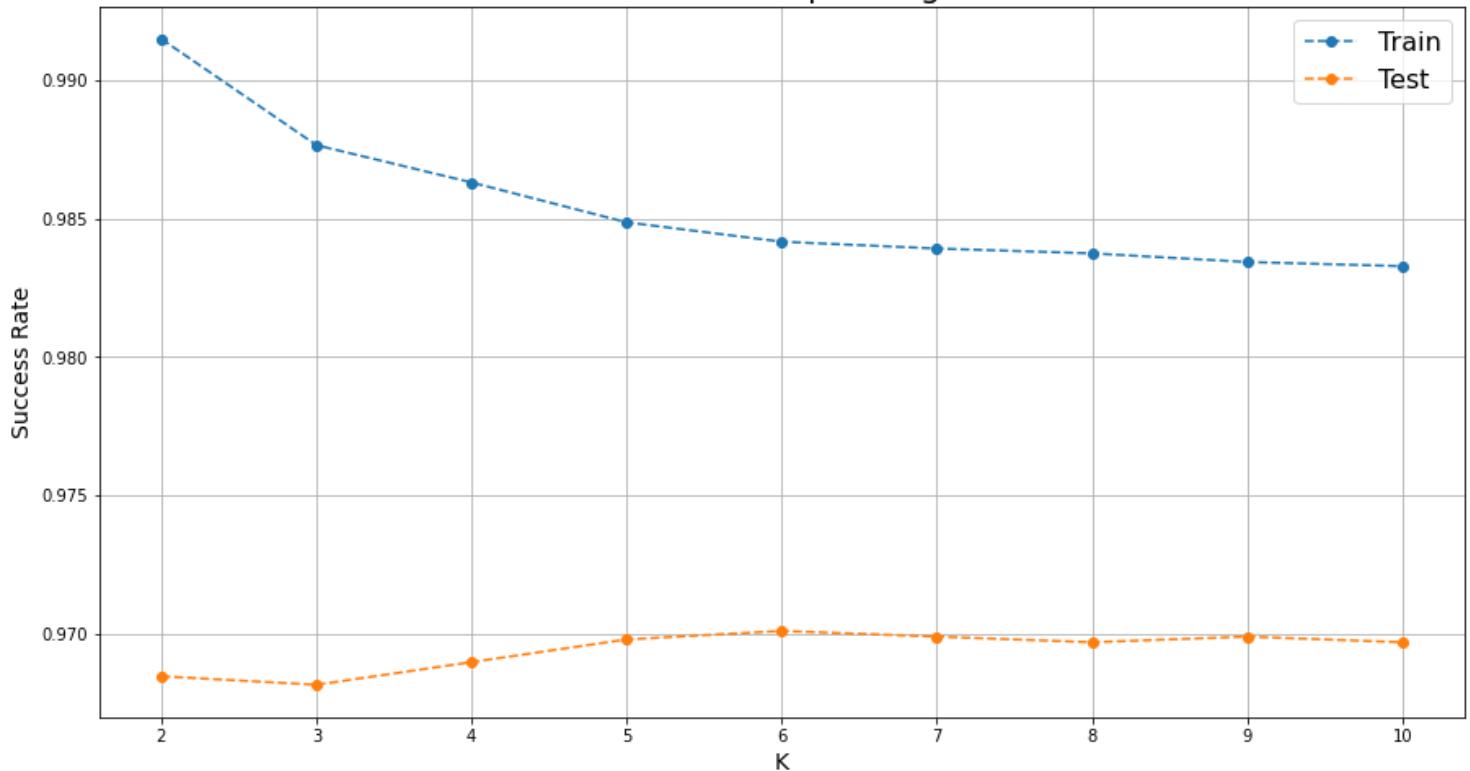
LDA 5.2

Size	Best K - Test	Test Accuracy	Best K - Train	Train Accuracy
32x32	6	97.009	2	99.15
64x64	10	97.347	2	100
128x128	3	70.73	ALL	100

טבלה 2: תוצאות LDA

32 × 32 5.2.1

Success Rate Depending On K



Train (Includes test results)

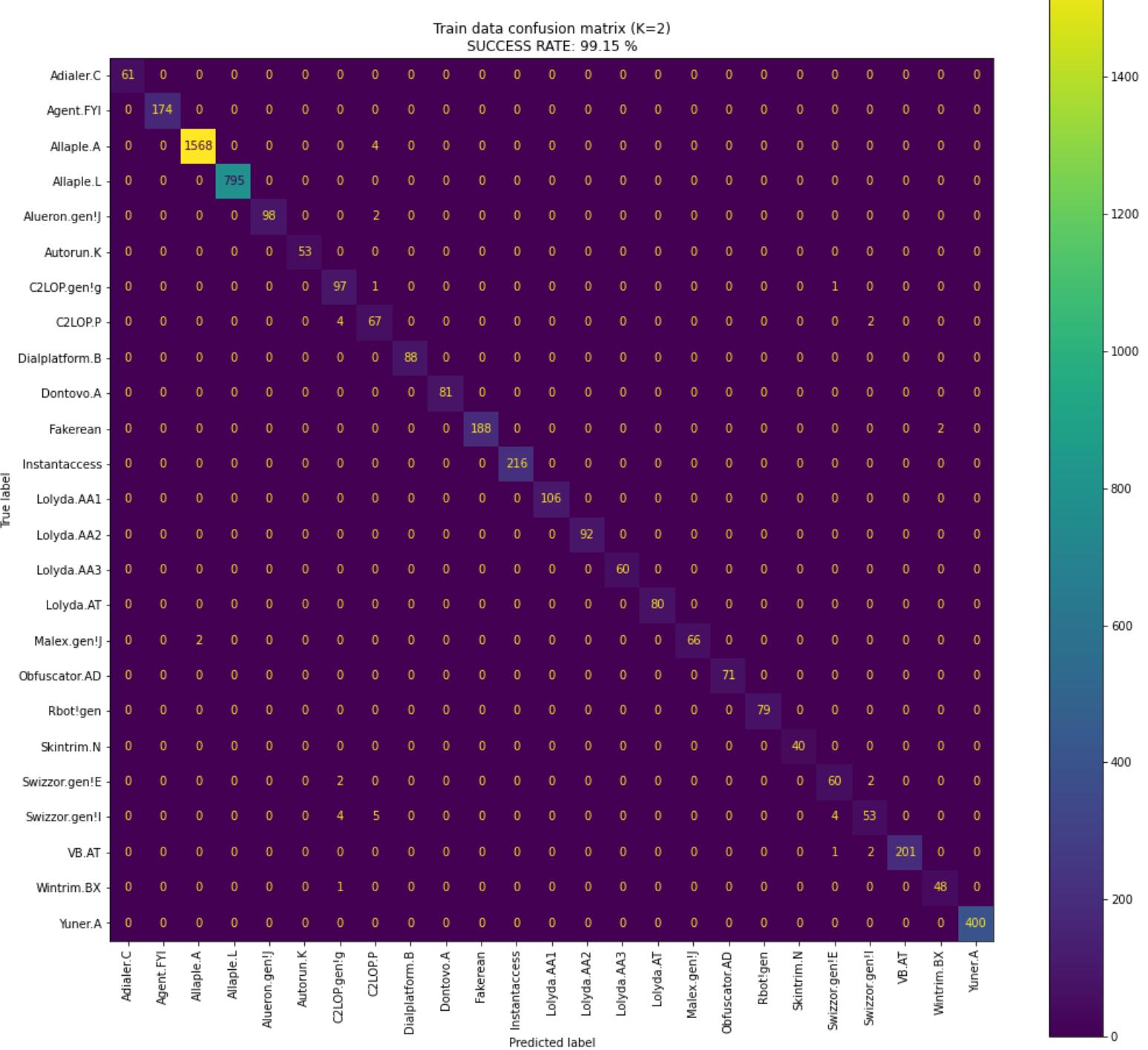
K = 2

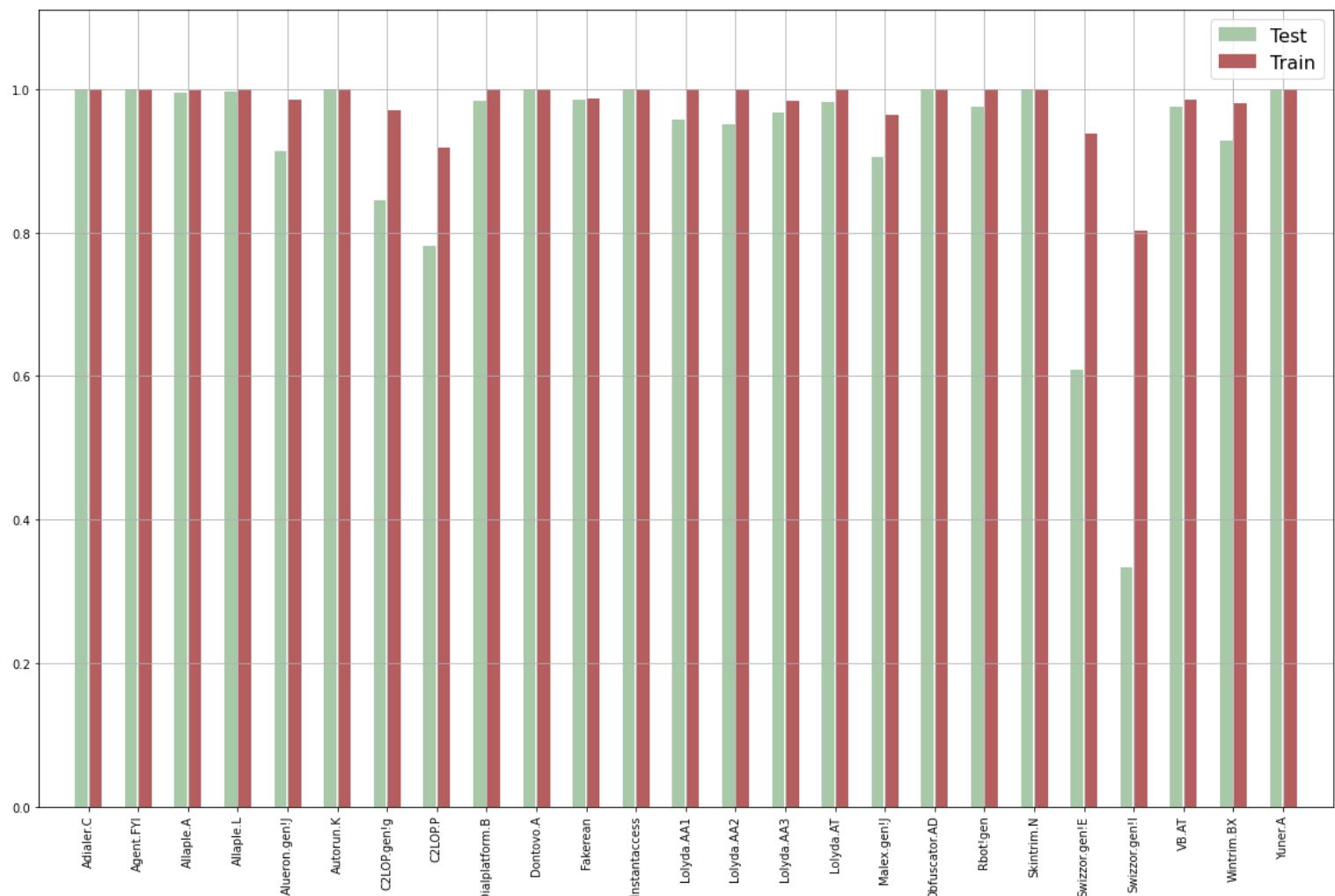
Success Rate (after normalizing):

- Train: 99.150 %
- Test: 96.846 %

Malware family with the highest false classification (after normalizing):

- Train: Swizzor.gen!I
- Test: Swizzor.gen!I





Test (Includes train results)

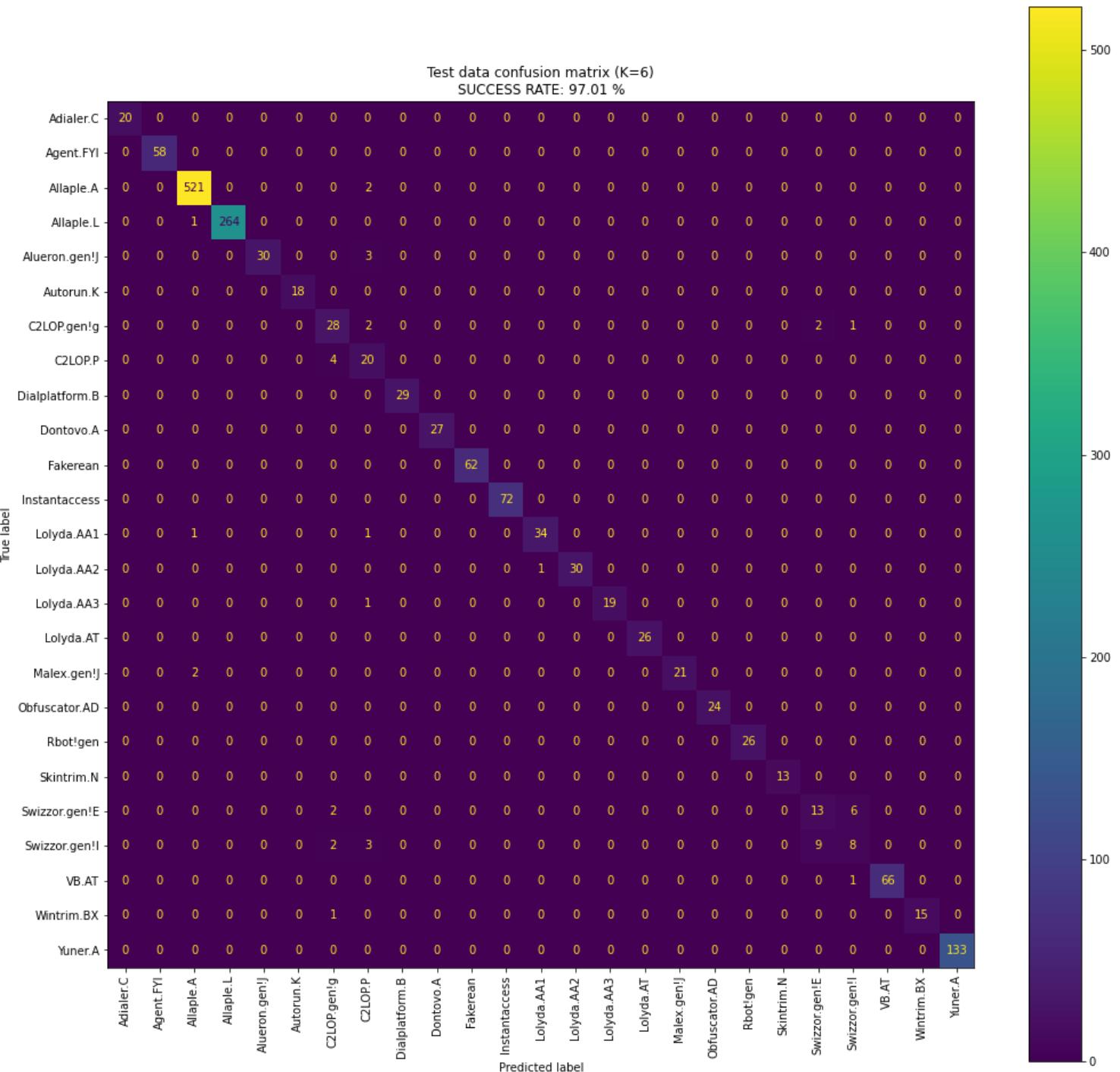
K = 6

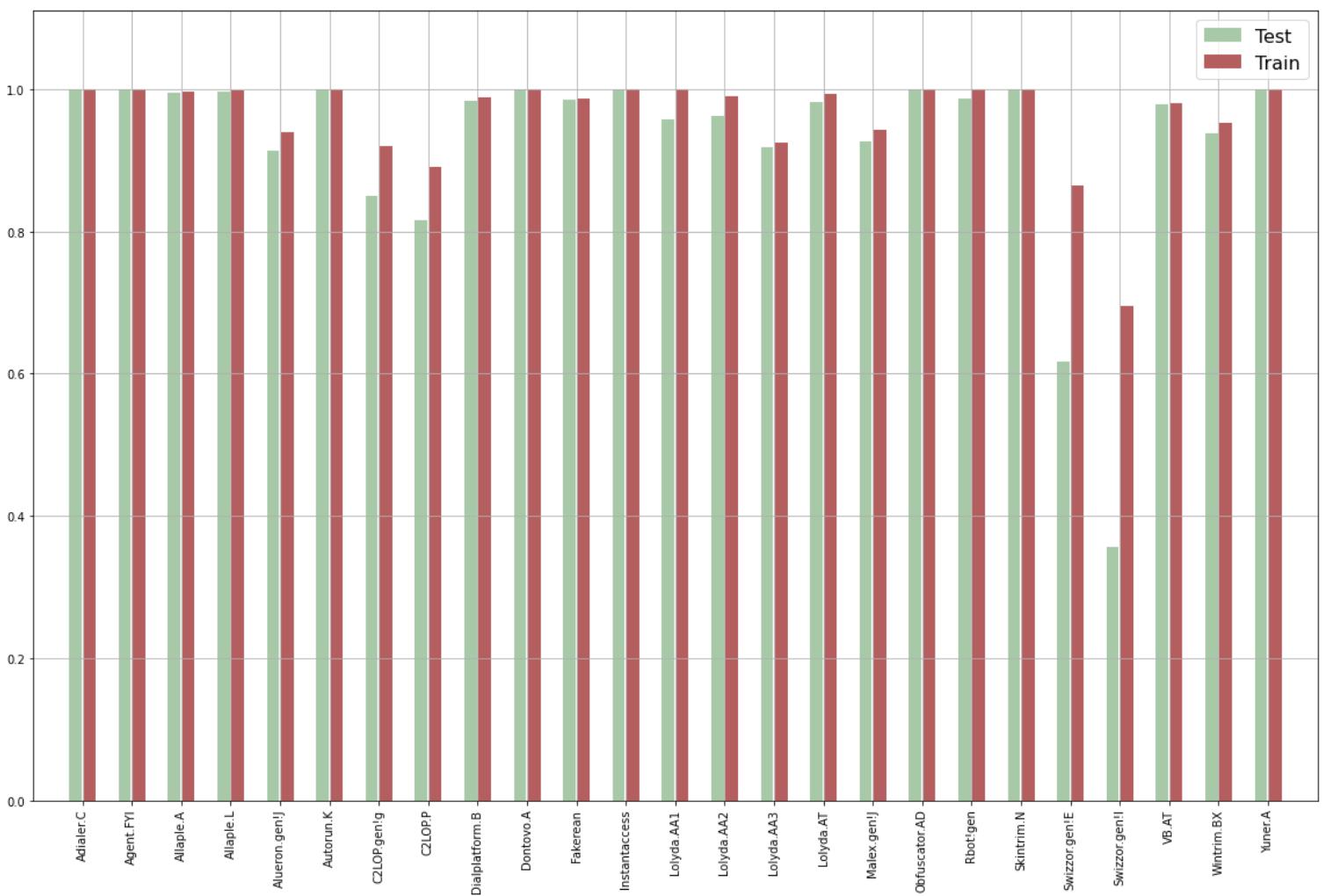
Success Rate (after normalizing):

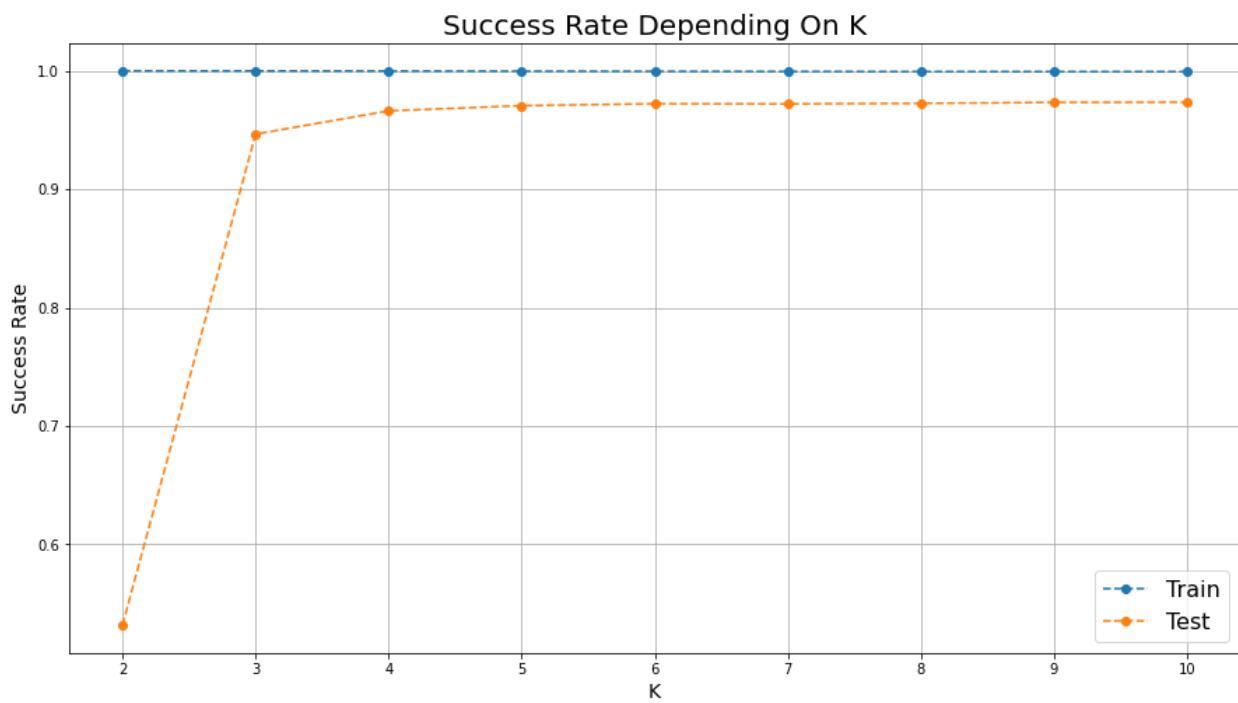
- Train: 98.417 %
- Test: 97.009 %

Malware family with the highest false classification (after normalizing):

- Train: Swizzor.gen!I
- Test: Swizzor.gen!I







Train (Includes test results)

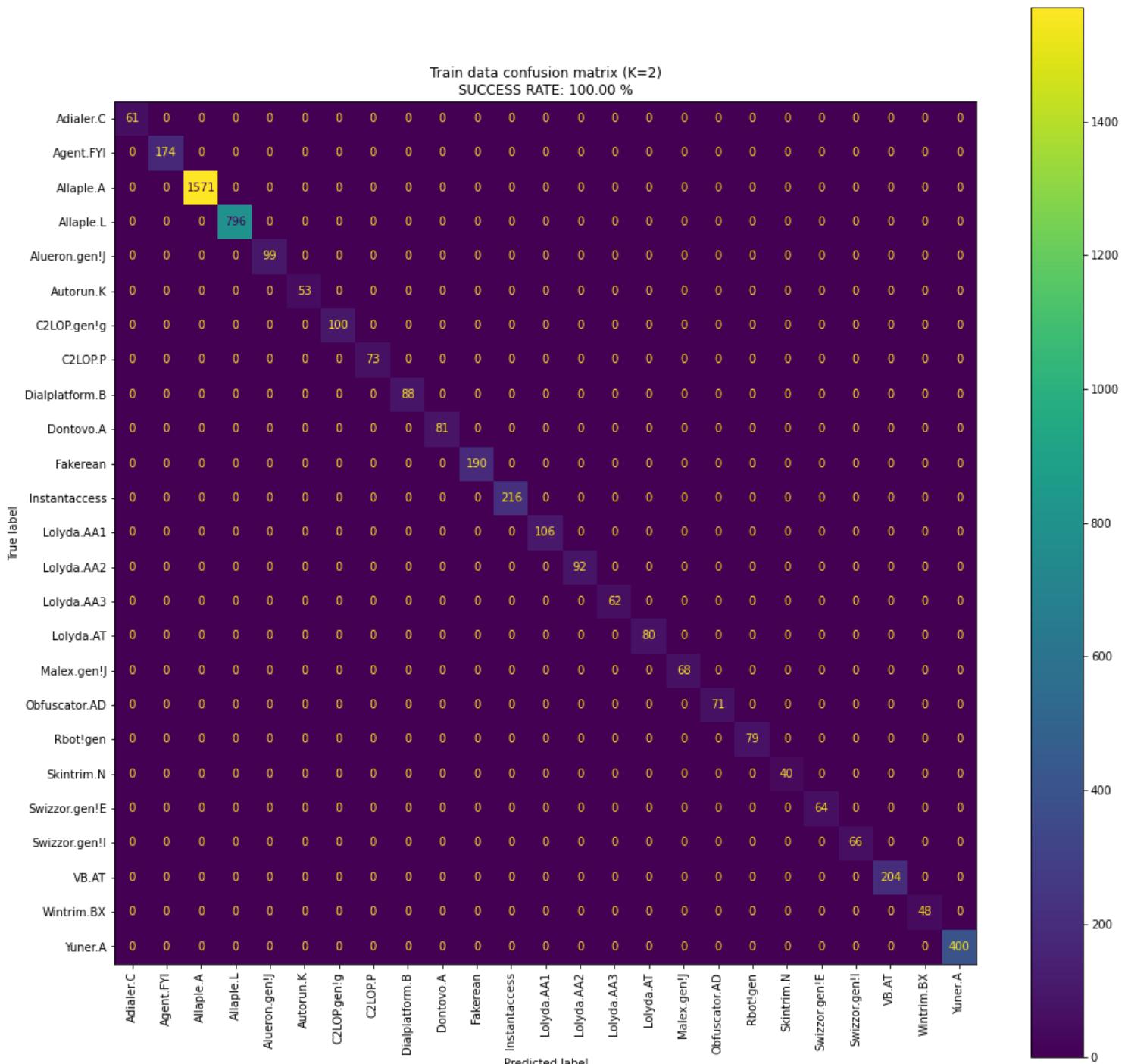
K = 2

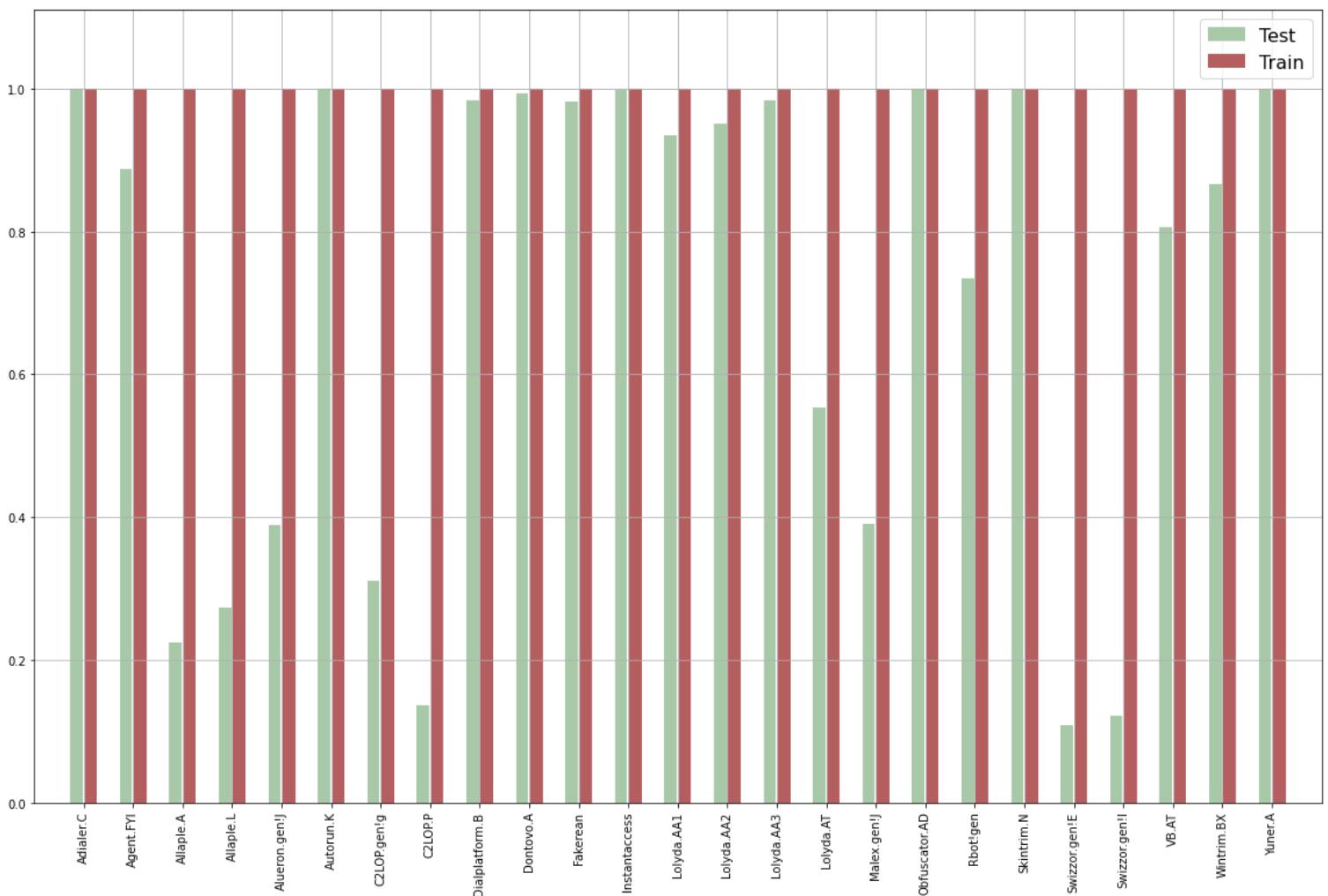
Success Rate (after normalizing):

- Train: 100.000 %
- Test: 53.175 %

Malware family with the highest false classification (after normalizing):

- Train: N/A
- Test: Swizzor.gen!E





Test (Includes train results)

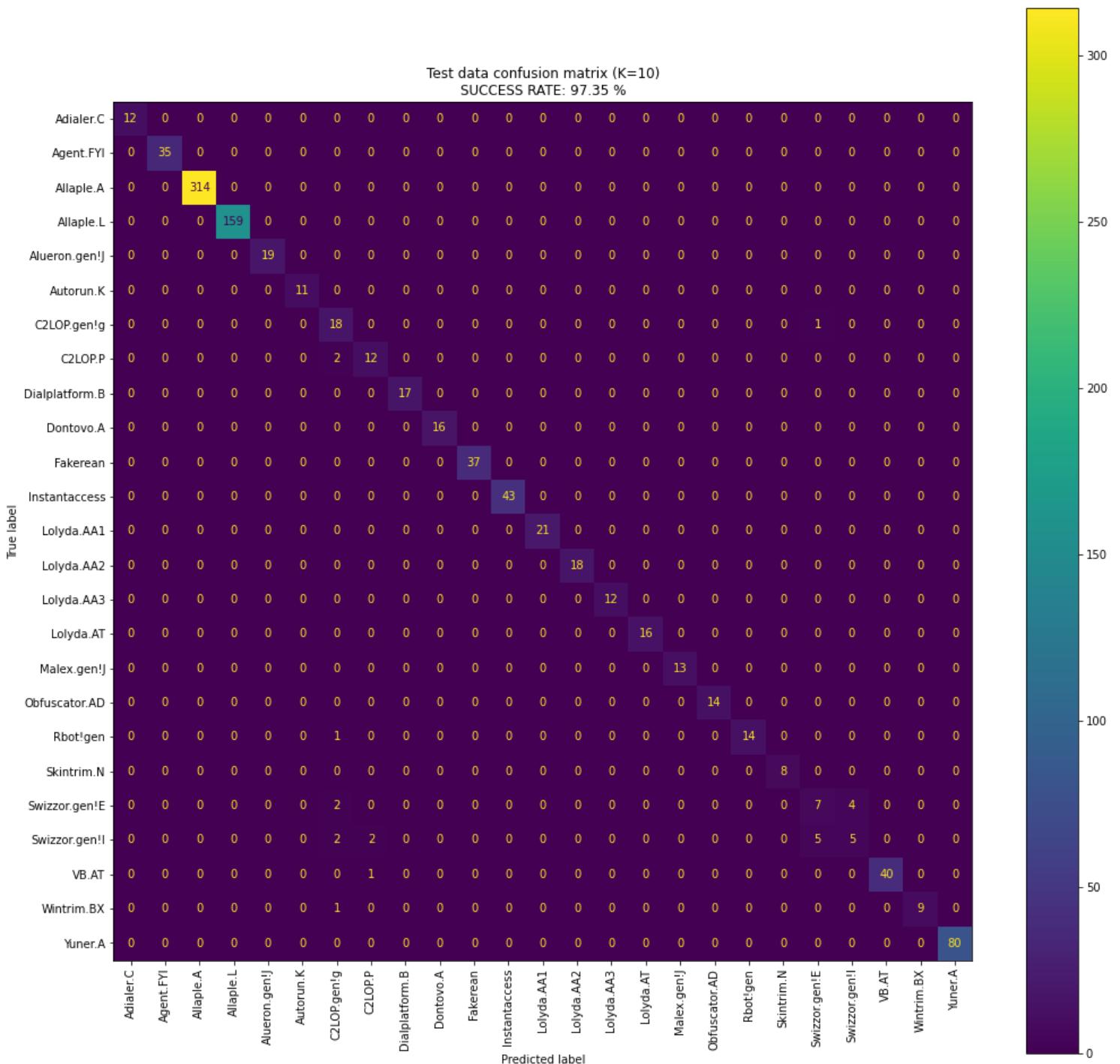
K = 10

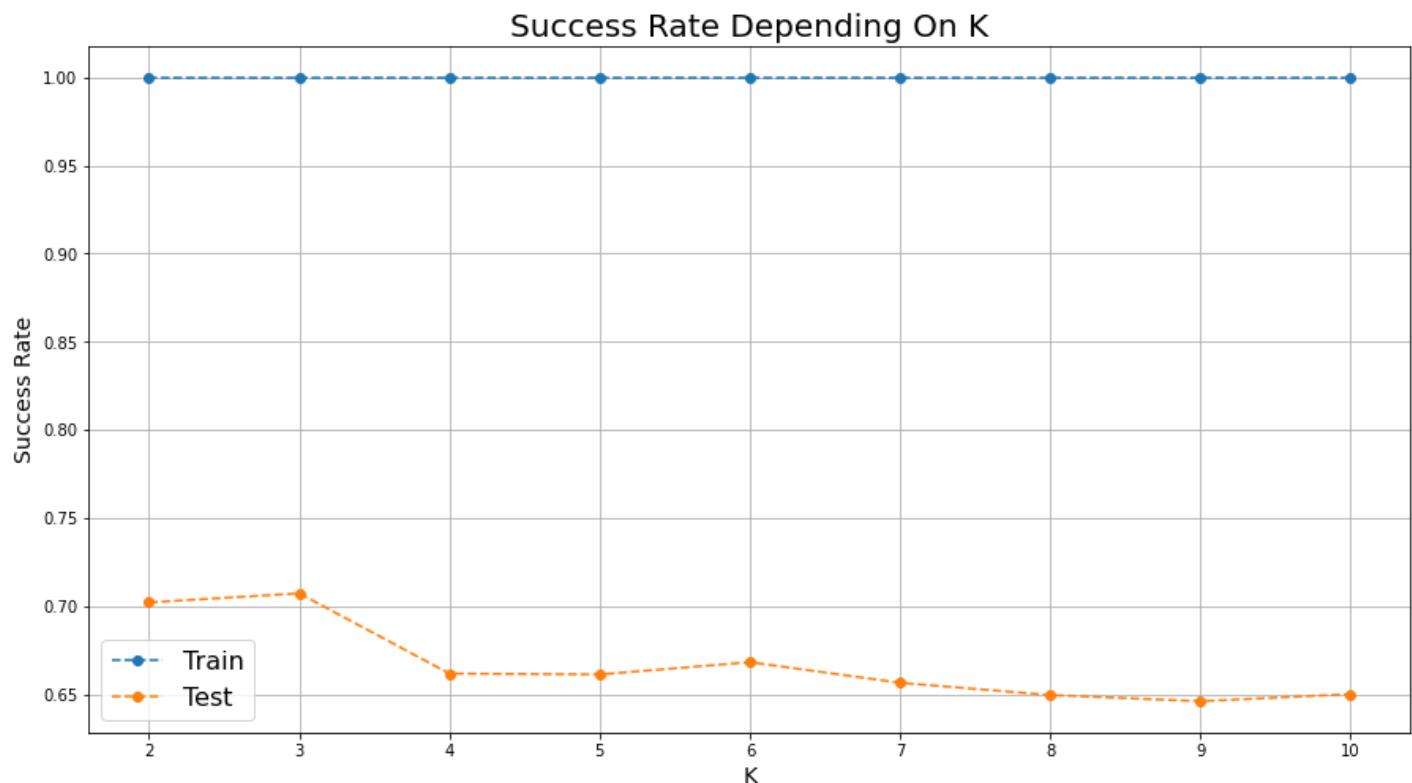
Success Rate (after normalizing):

- Train: 99.933 %
- Test: 97.347 %

Malware family with the highest false classification (after normalizing):

- Train: Lolyda.AA3
- Test: Swizzor.gen!I





Train (Includes test results)

K = ALL

Success Rate (after normalizing):

- Train: 100.000 %

Malware family with the highest false classification (after normalizing): N/A

Test (Includes train results)

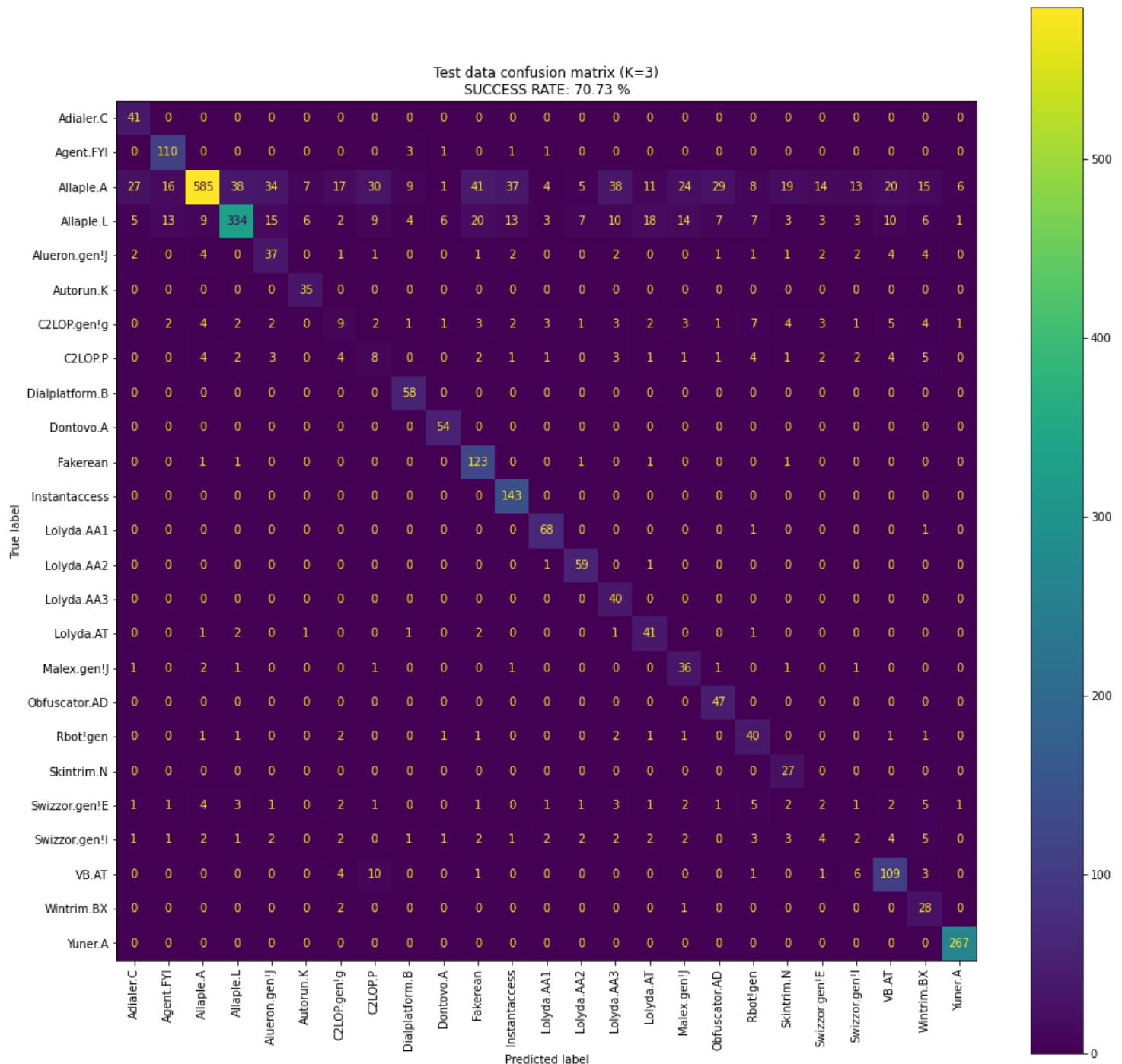
K = 3

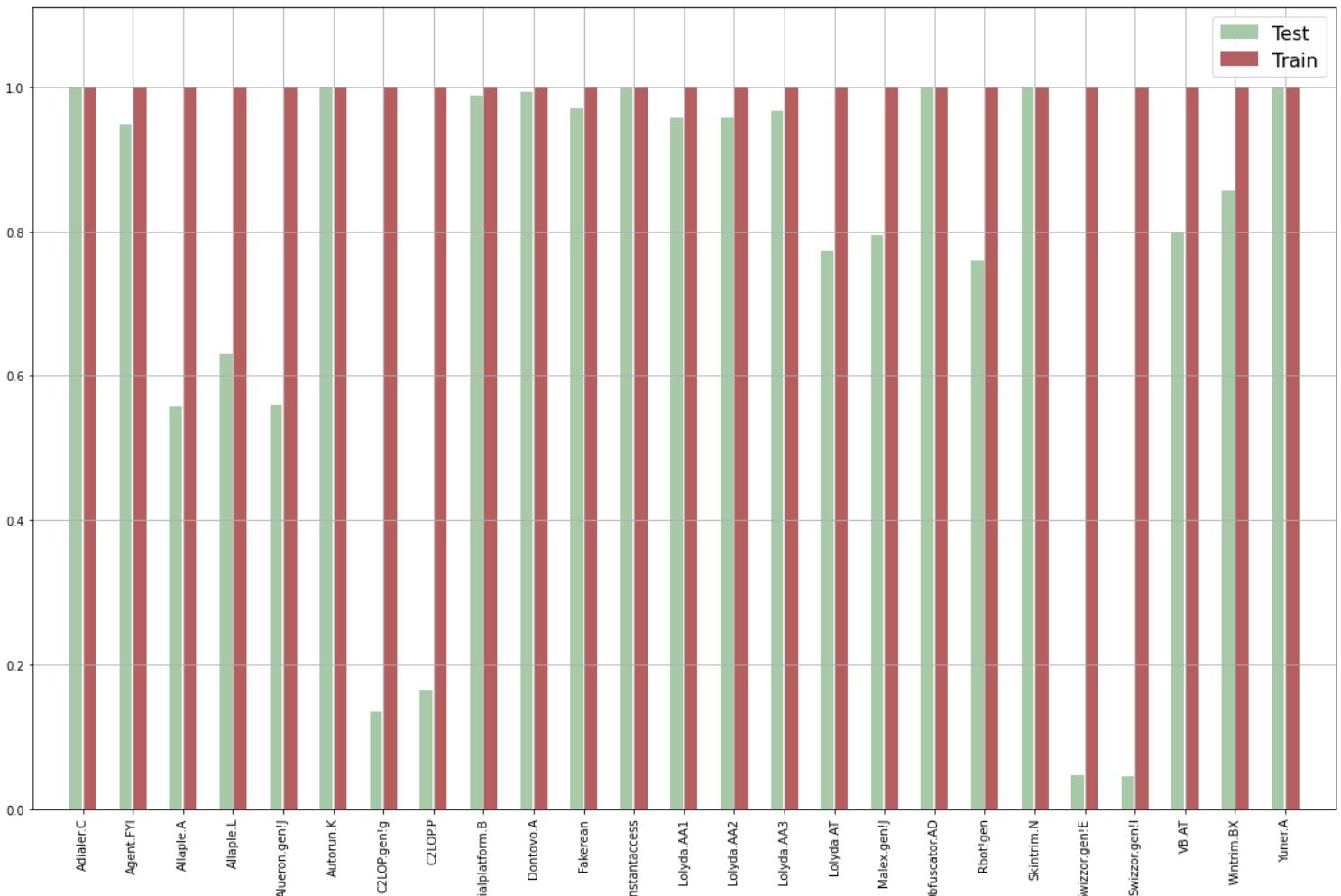
Success Rate (after normalizing):

- Train: 100.000 %
- Test: 70.730 %

Malware family with the highest false classification (after normalizing):

- Train: N/A
- Test: Swizzor.gen!I

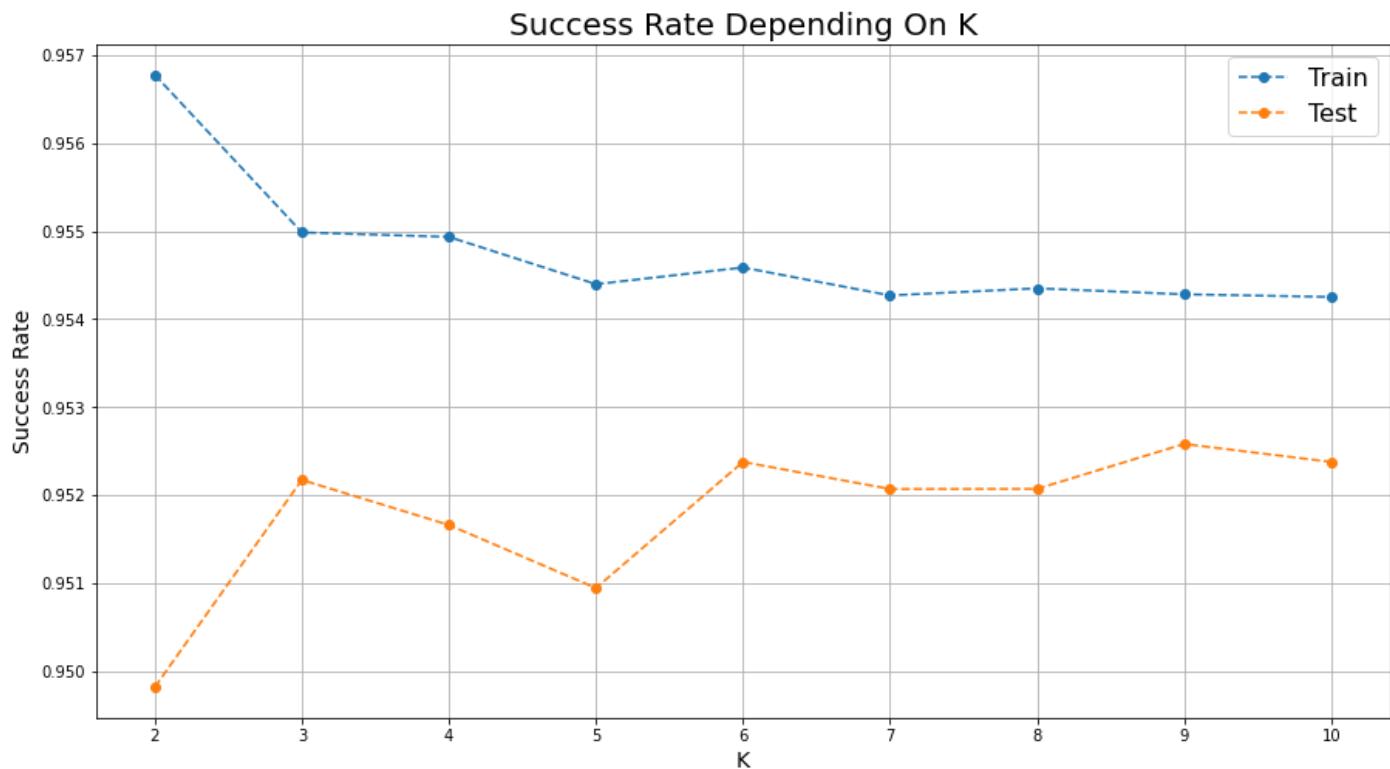




Size	Best K - Test	Test Accuracy	Best K - Train	Train Accuracy
32x32	9	95.258	2	95.678
64x64	3	96.467	2	97.111
128x128	3	96.887	2	97.634

טבלה 3: תוצאות GNB

32 × 32 5.3.1



Train (Includes test results)

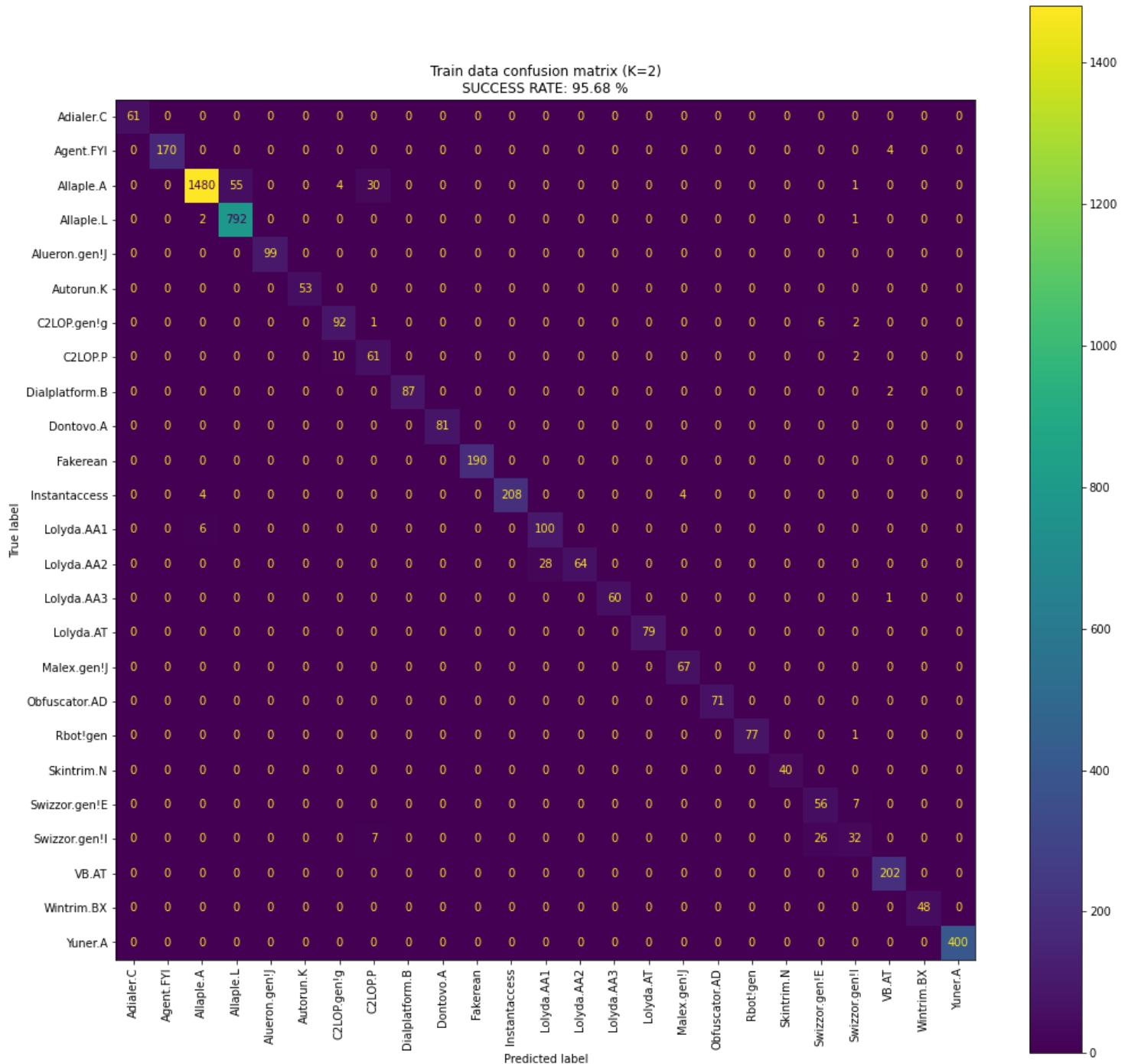
K = 2

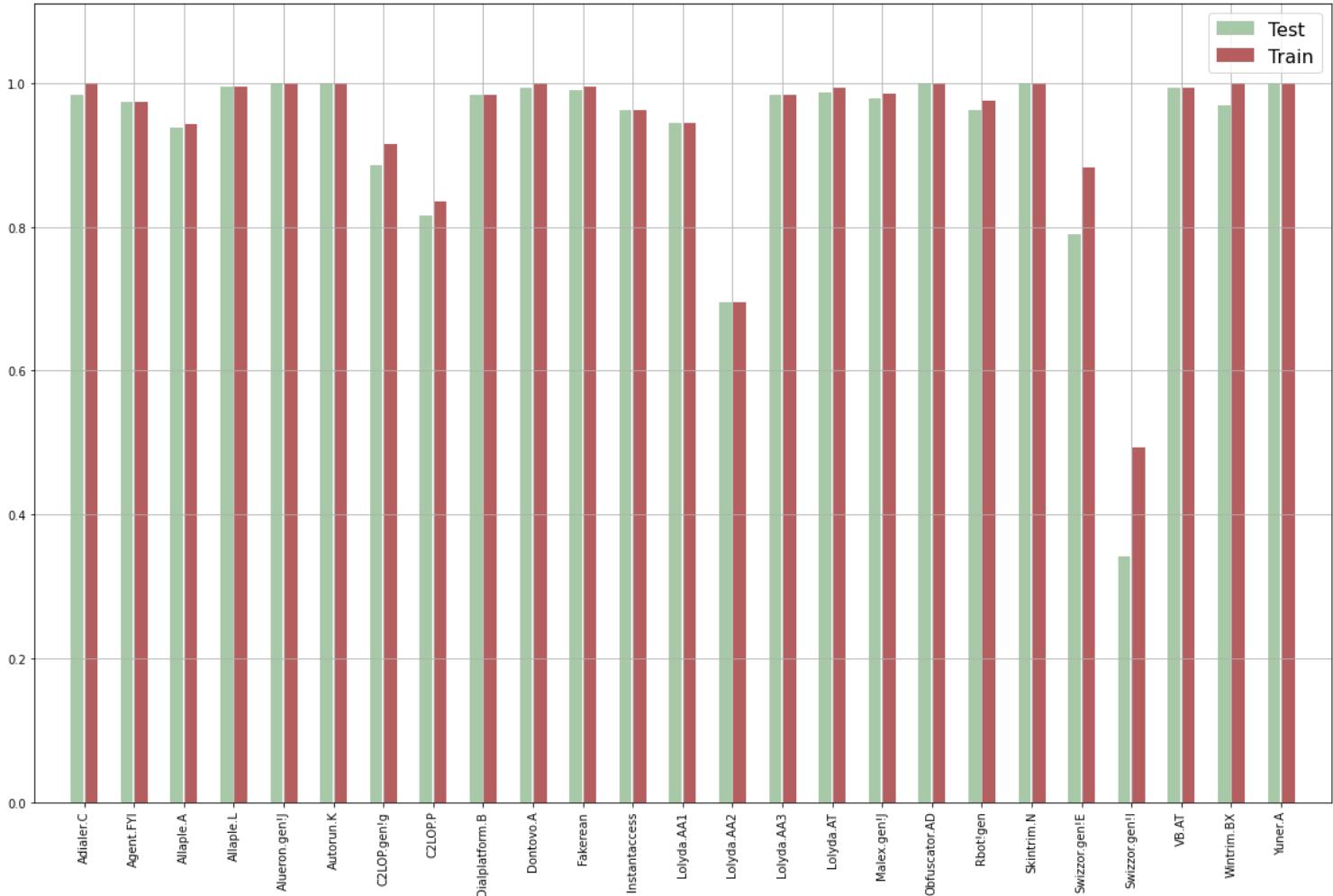
Success Rate (after normalizing):

- Train: 95.678 %
- Test: 94.982 %

Malware family with the highest false classification (after normalizing):

- Train: Swizzor.gen!I
- Test: Swizzor.gen!I





Test (Includes train results)

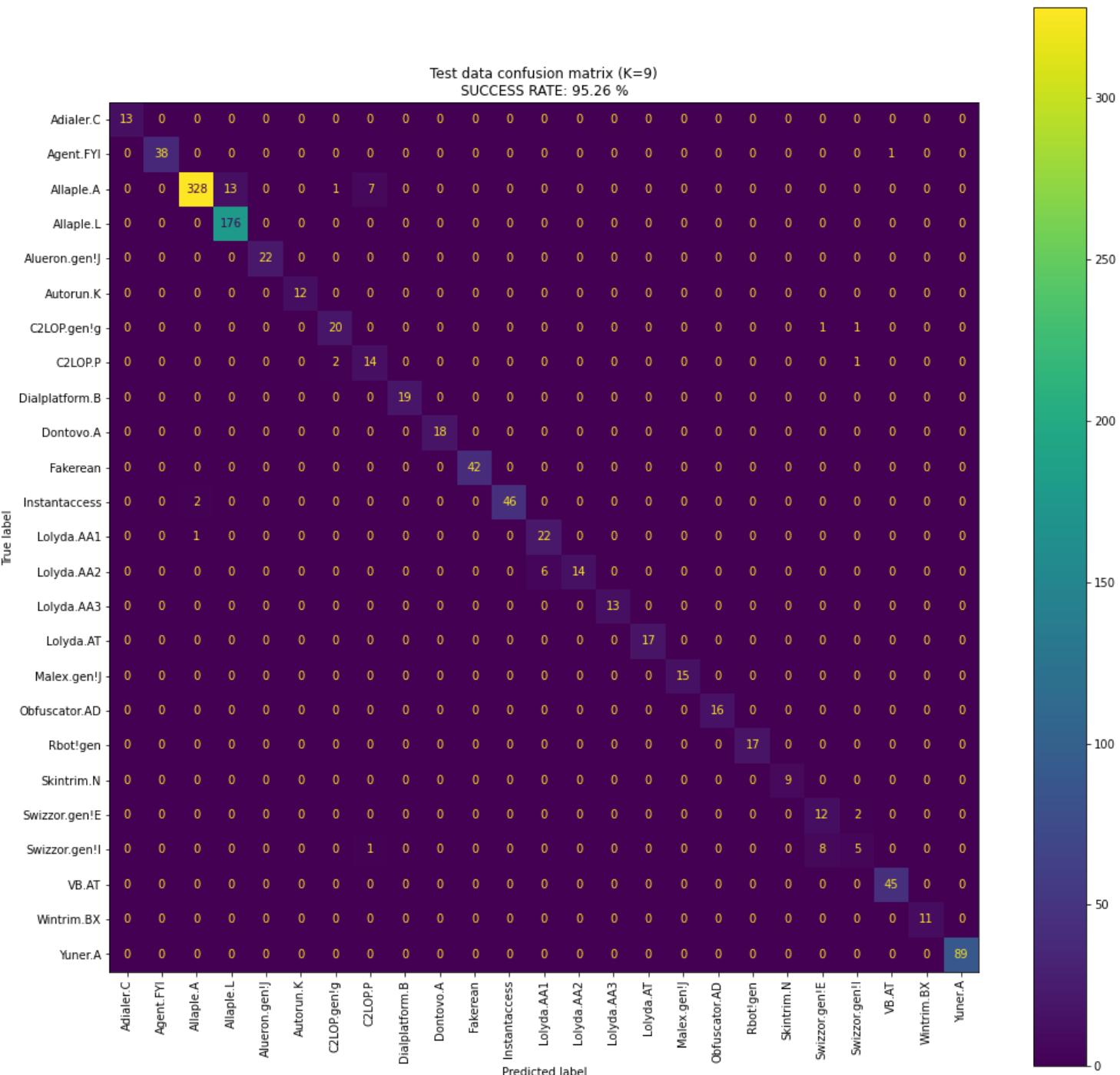
K = 9

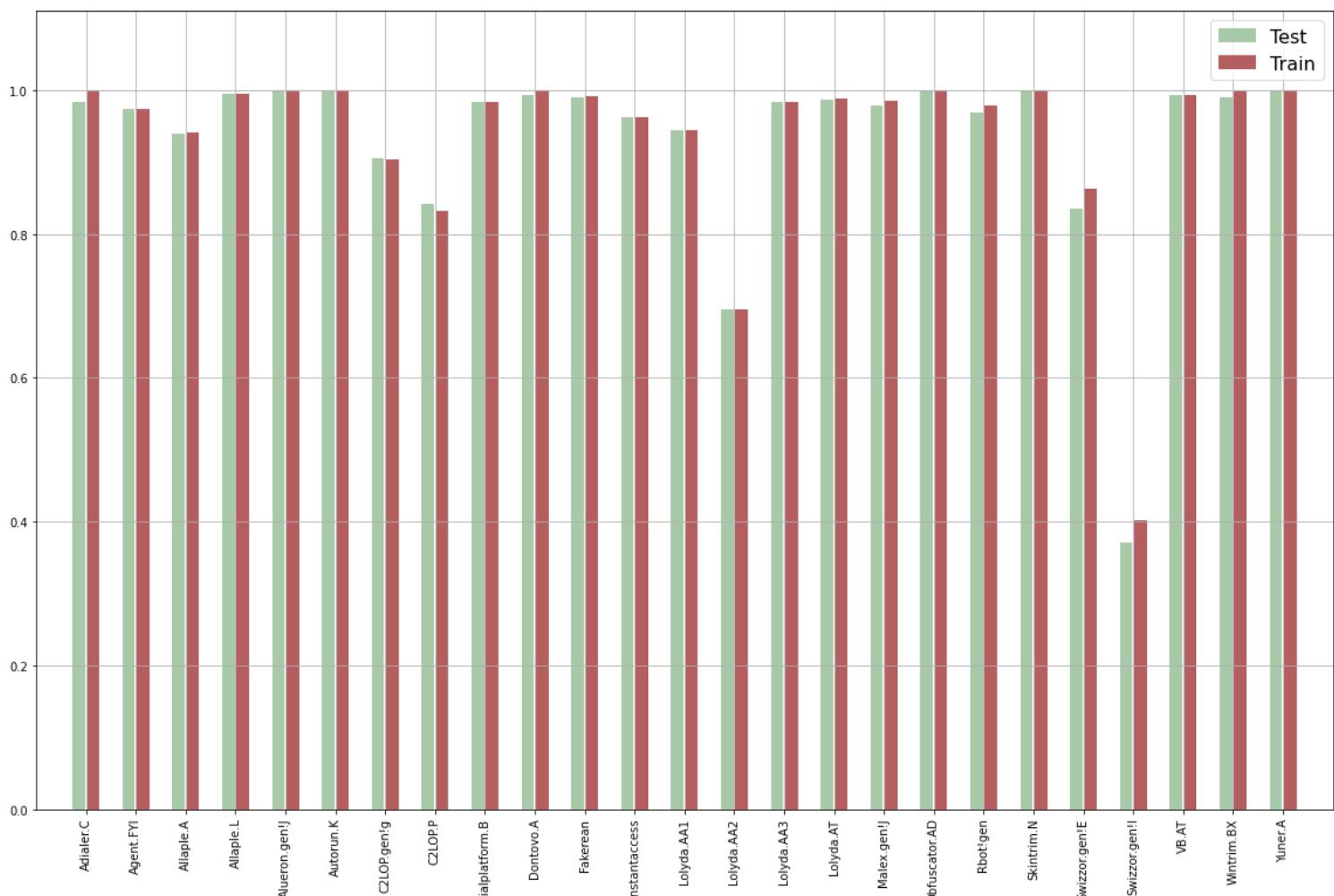
Success Rate (after normalizing):

- Train: 95.428 %
- Test: 95.258 %

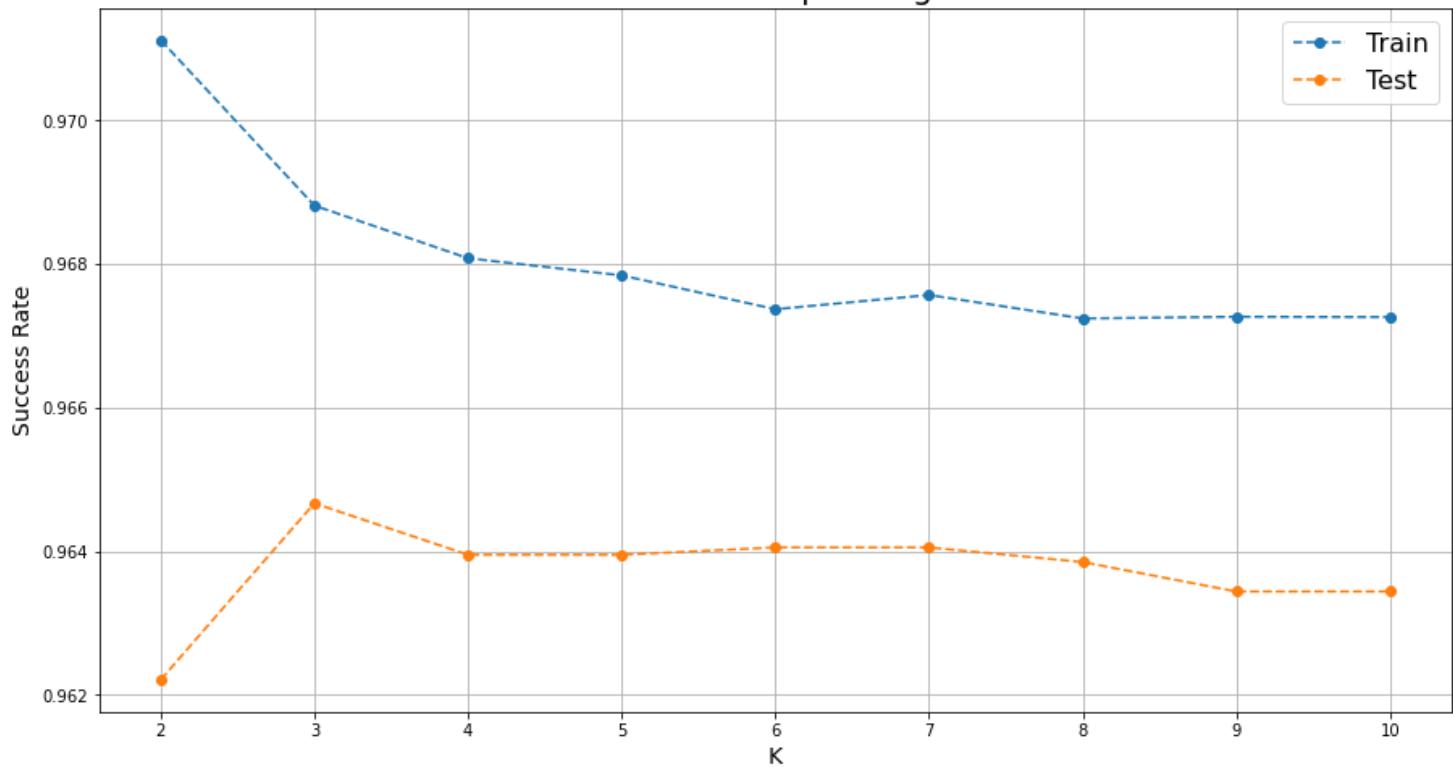
Malware family with the highest false classification (after normalizing):

- Train: Swizzor.gen!I
- Test: Swizzor.gen!I





Success Rate Depending On K



Train (Includes test results)

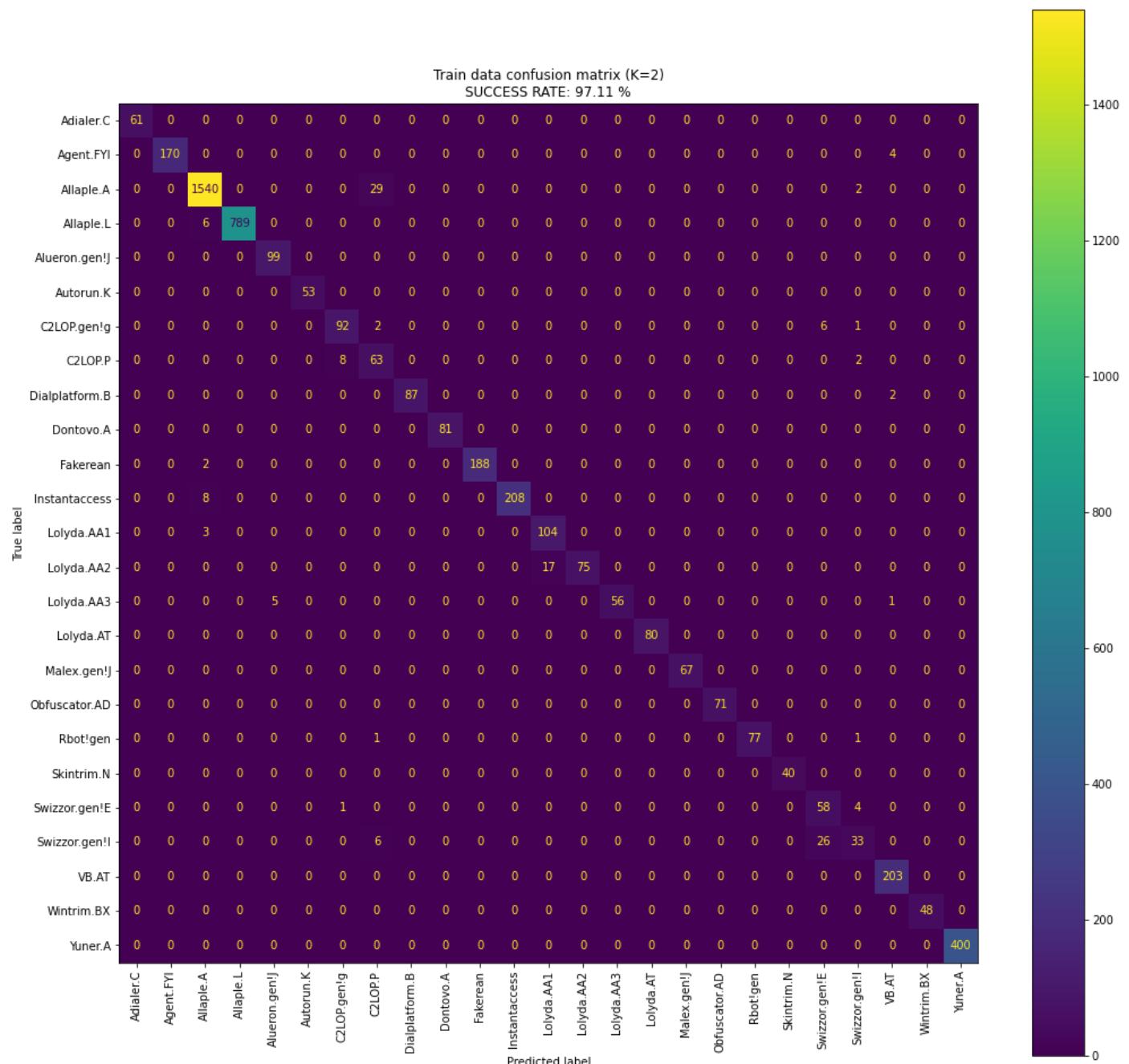
K = 2

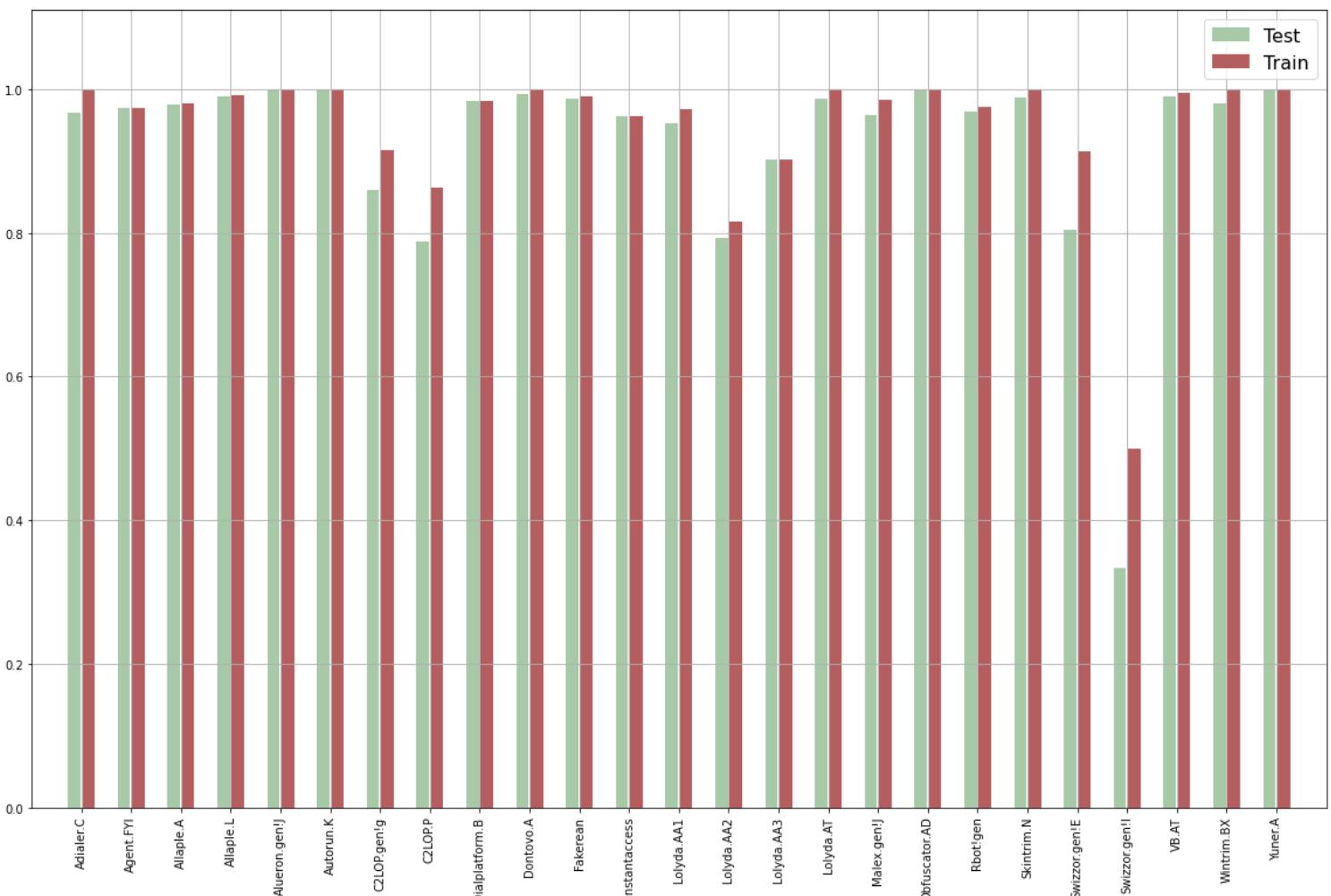
Success Rate (after normalizing):

- Train: 97.112 %
- Test: 96.221 %

Malware family with the highest false classification (after normalizing):

- Train: Swizzor.gen!I
- Test: Swizzor.gen!I





Test (Includes train results)

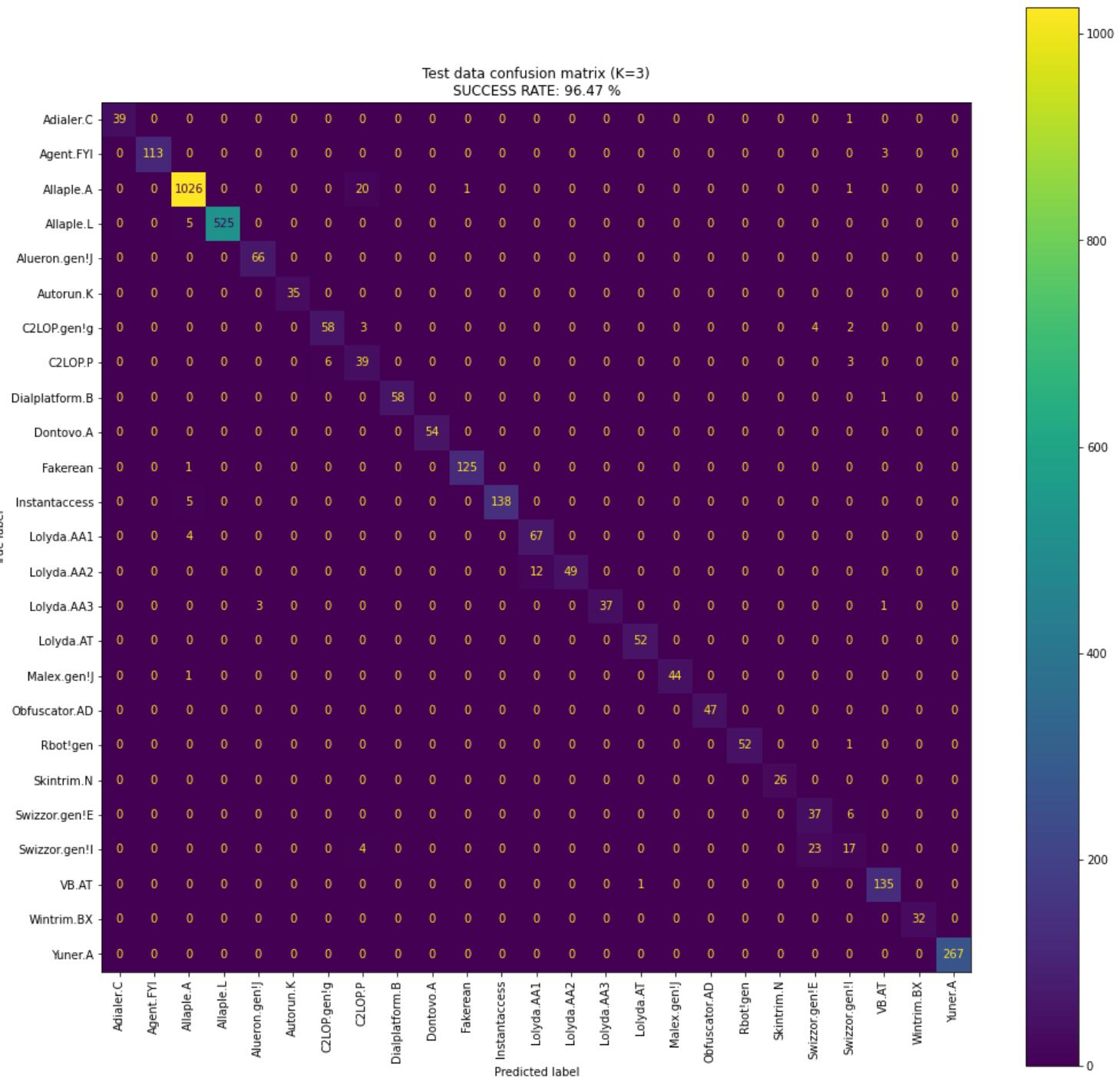
K = 3

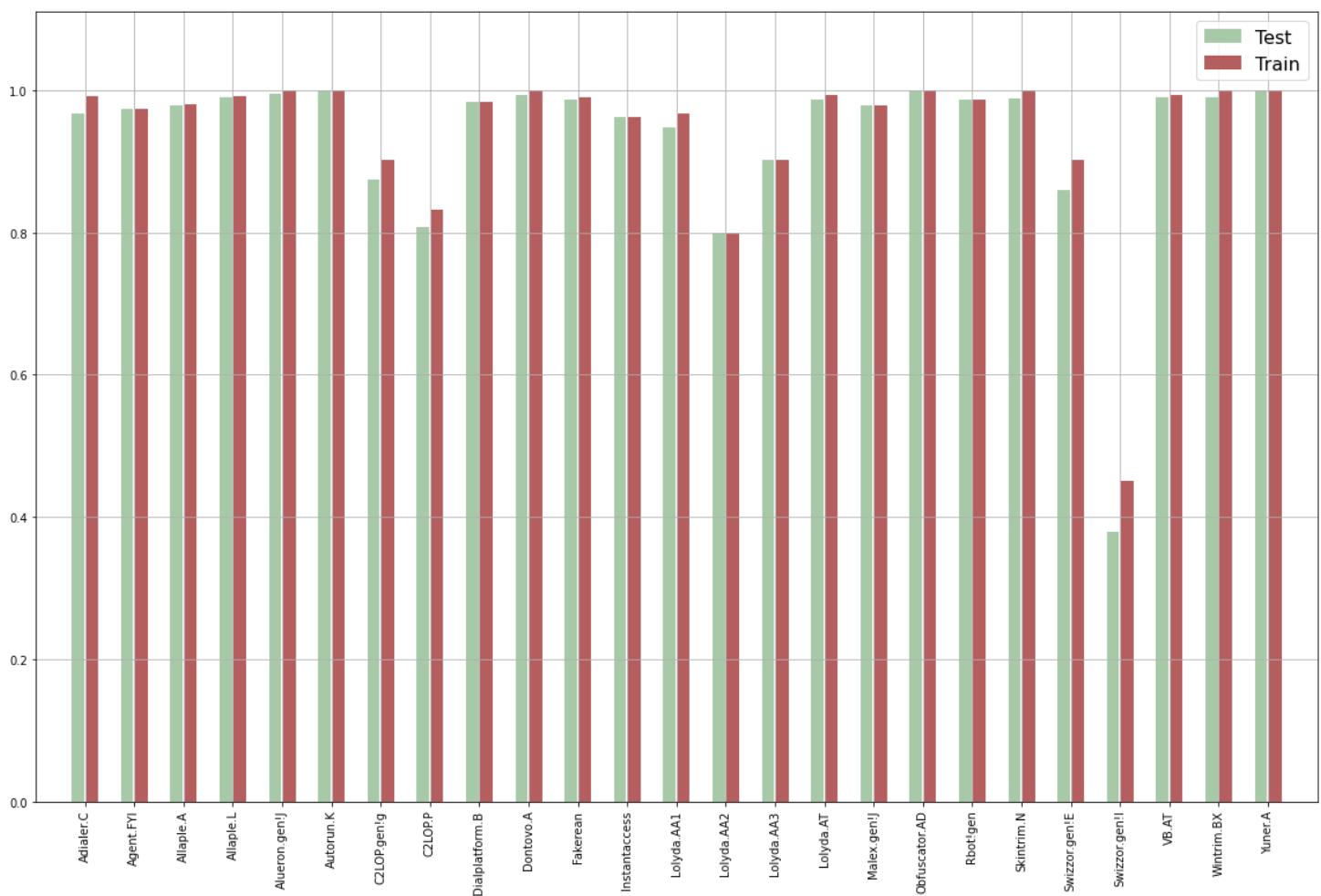
Success Rate (after normalizing):

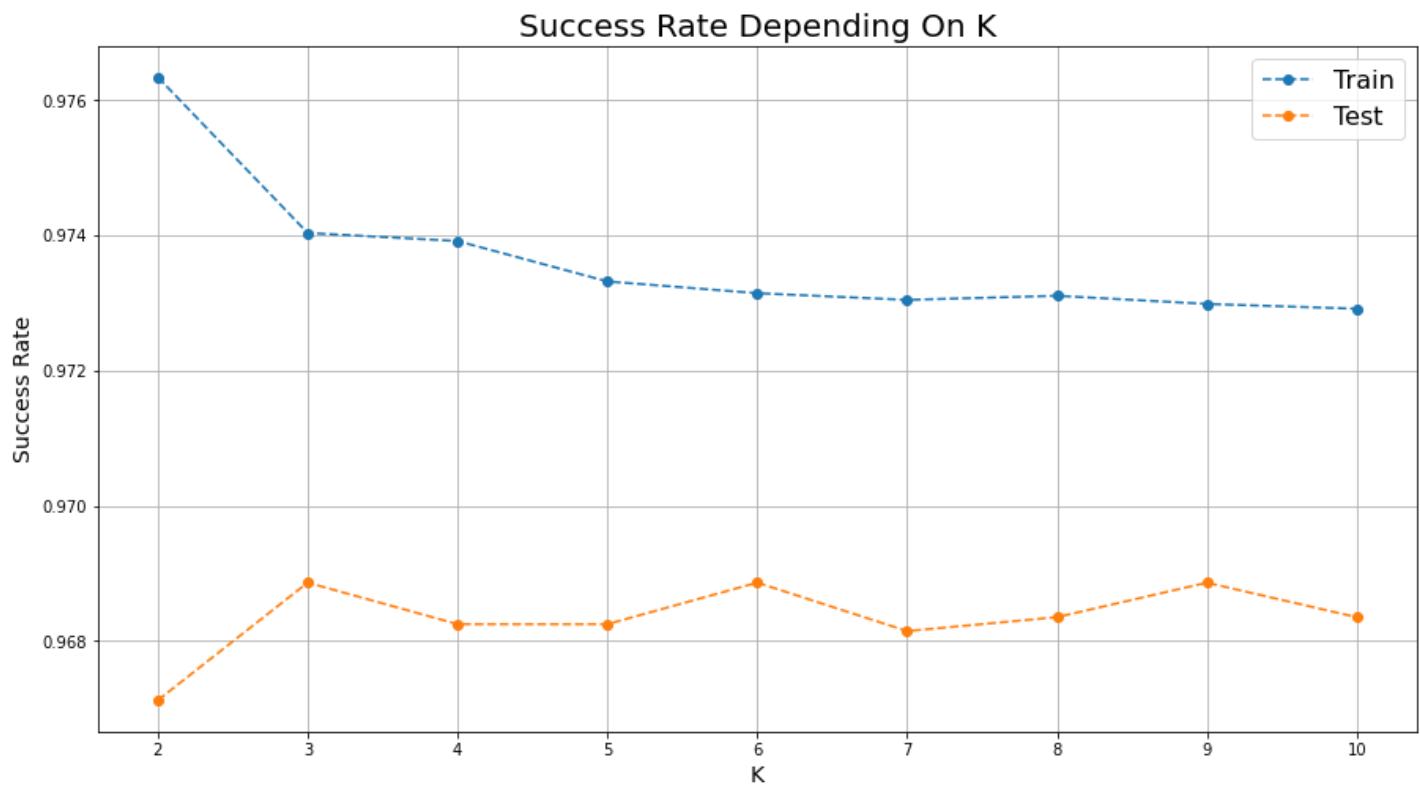
- Train: 96.881 %
- Test: 96.467 %

Malware family with the highest false classification (after normalizing):

- Train: Swizzor.gen!I
- Test: Swizzor.gen!I







Train (Includes test results)

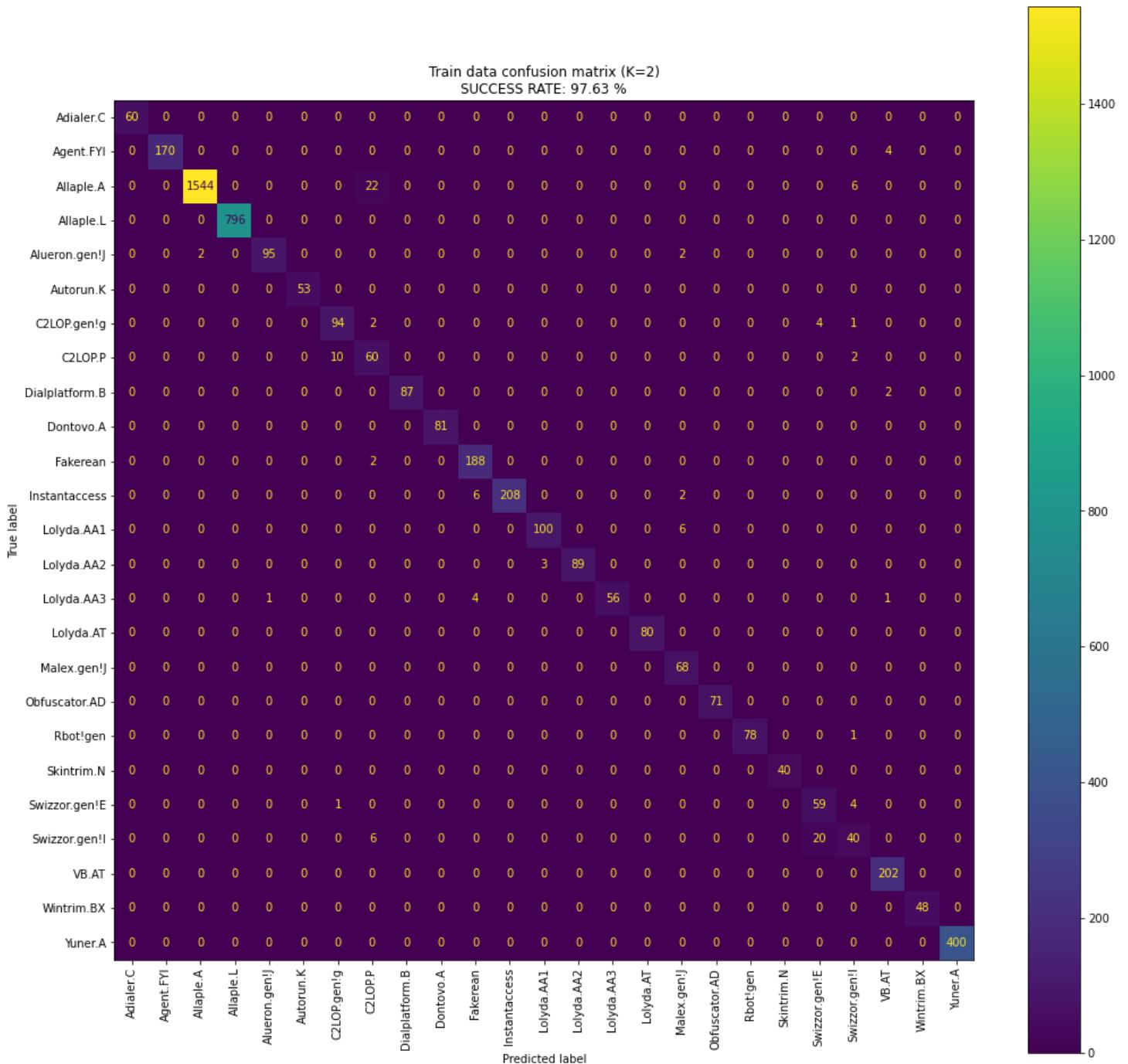
K = 2

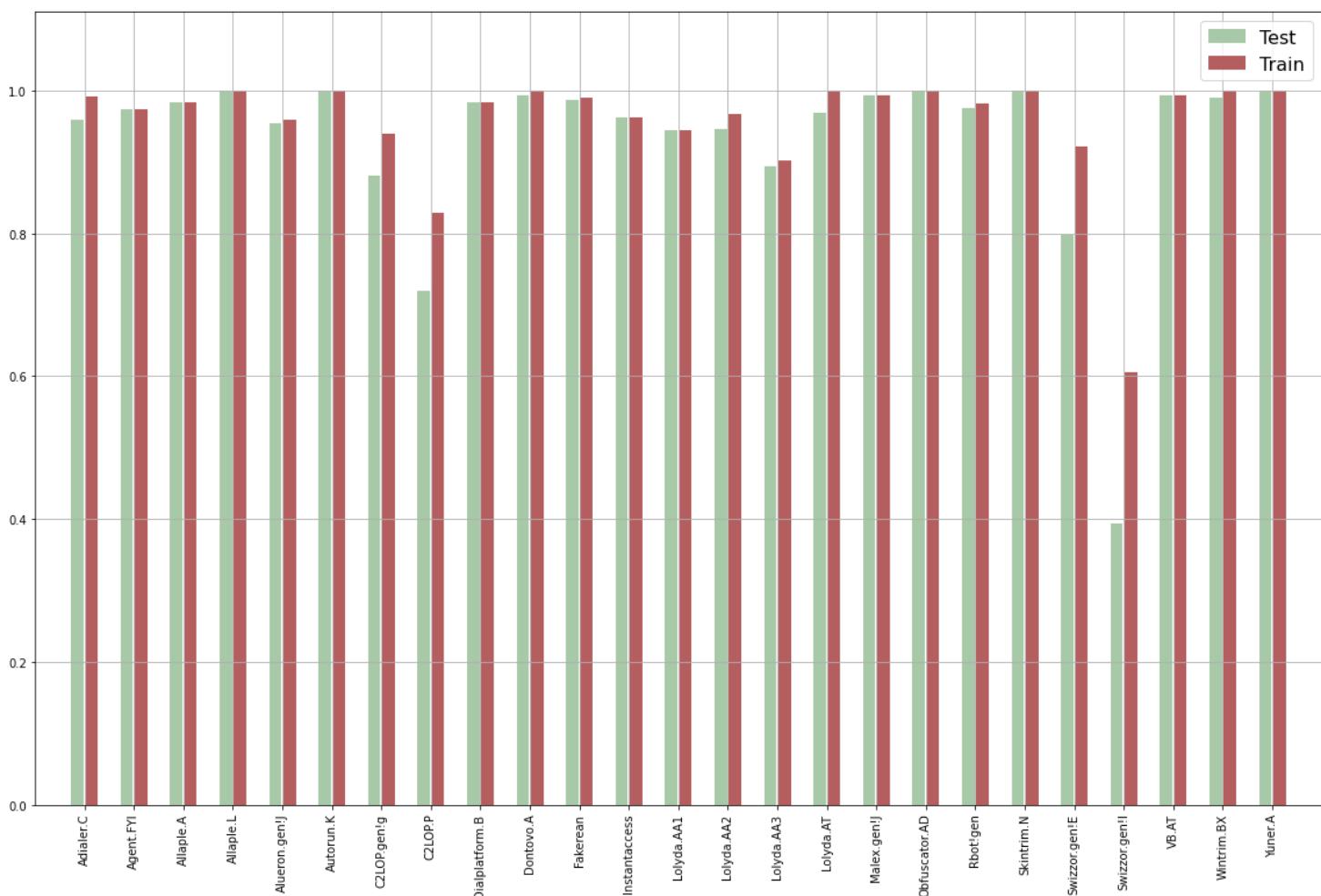
Success Rate (after normalizing):

- Train: 97.634 %
- Test: 96.712 %

Malware family with the highest false classification (after normalizing):

- Train: Swizzor.gen!I
- Test: Swizzor.gen!I





Test (Includes train results)

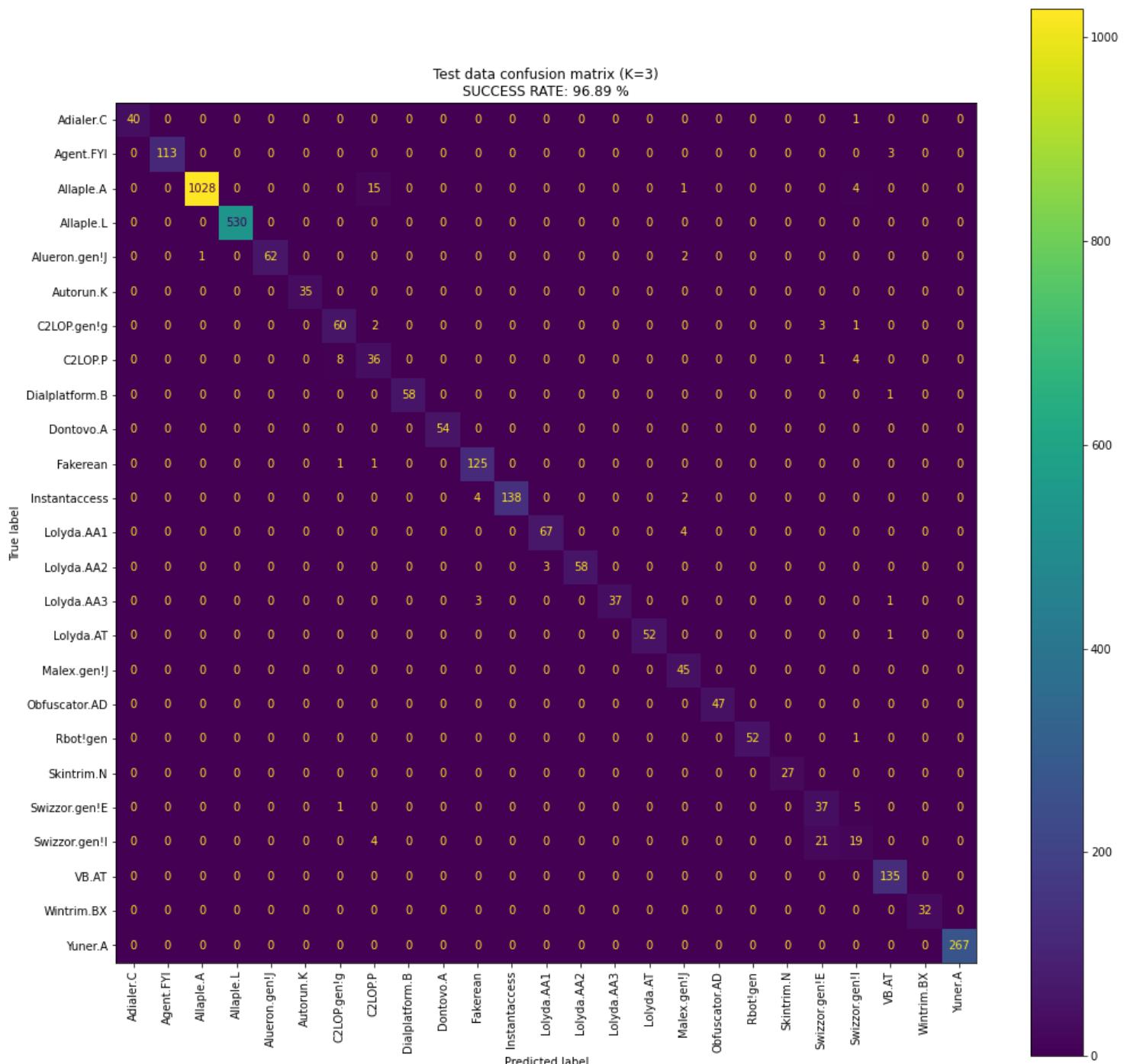
K = 3

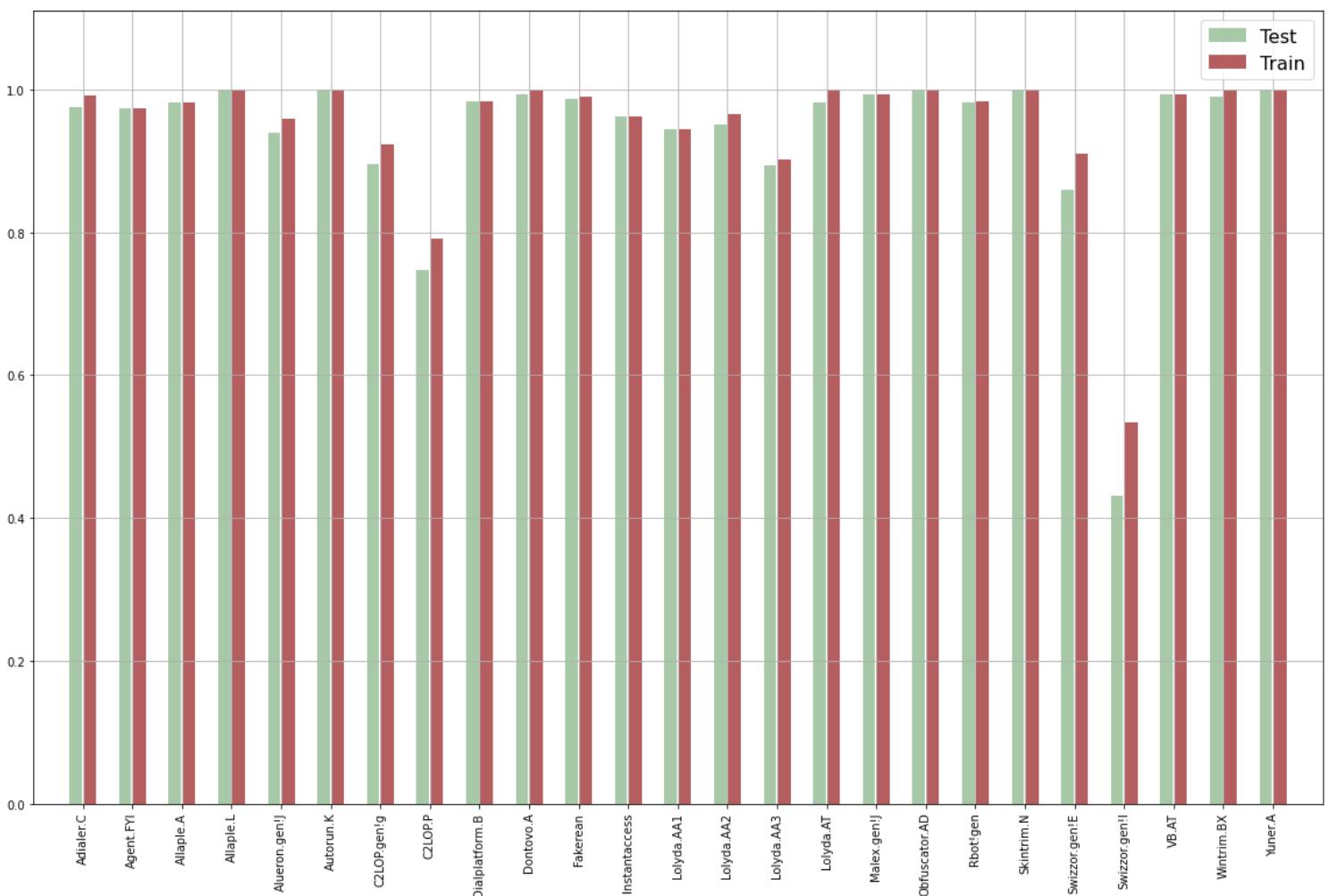
Success Rate (after normalizing):

- Train: 97.404 %
- Test: 96.887 %

Malware family with the highest false classification (after normalizing):

- Train: Swizzor.gen!I
- Test: Swizzor.gen!I





CNN 5.4
 מודל CNN שכבה 1, 32 פילטרים 5.4.1

CNN (1 Layer, 32 Filter, 11 Epochs)				
Size	Best K - Test	Test Accuracy	Best K - Train	Train Accuracy
32x32	8	98.71	8	99.188
64x64	8	99.754	8	99.997
128x128	10	99.816	10	100

טבלה 4: תוצאות CNN שכבה 1, 32 פילטרים

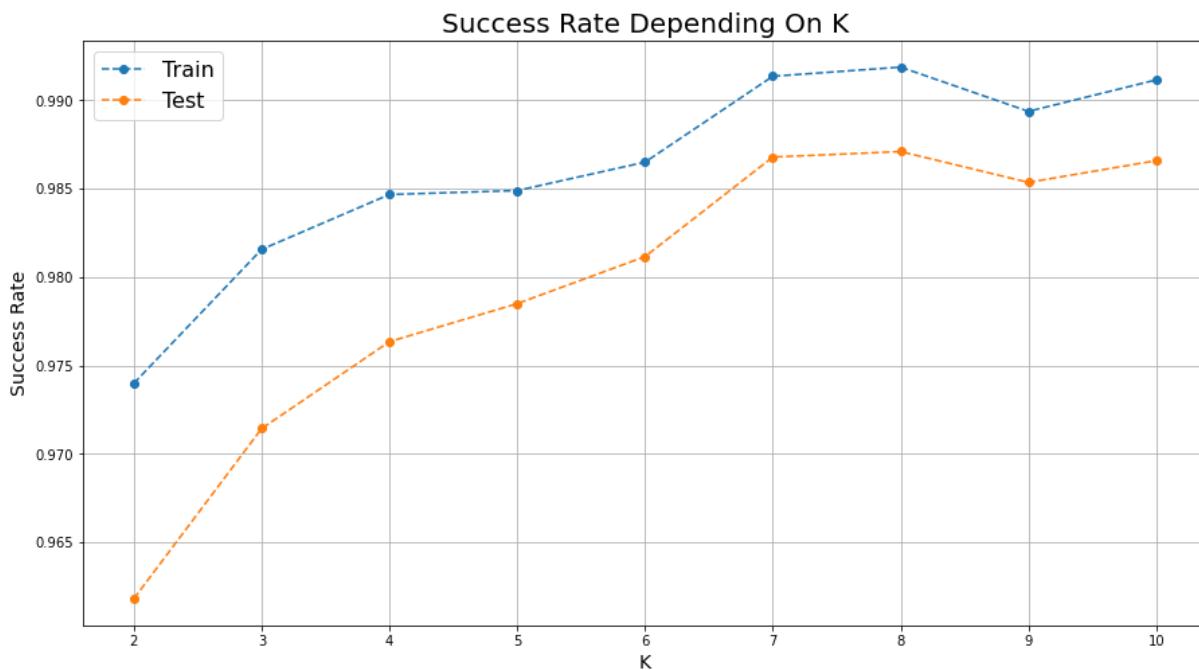
32×32 5.4.1.1

Model Summary:

Model: "sequential"

Layer (type)	Output Shape	Param #
conv2d (Conv2D)	(None, 32, 32, 32)	320
flatten (Flatten)	(None, 32768)	0
dense (Dense)	(None, 25)	819225

Total params: 819,545
 Trainable params: 819,545
 Non-trainable params: 0



Train & Test (Includes test results)

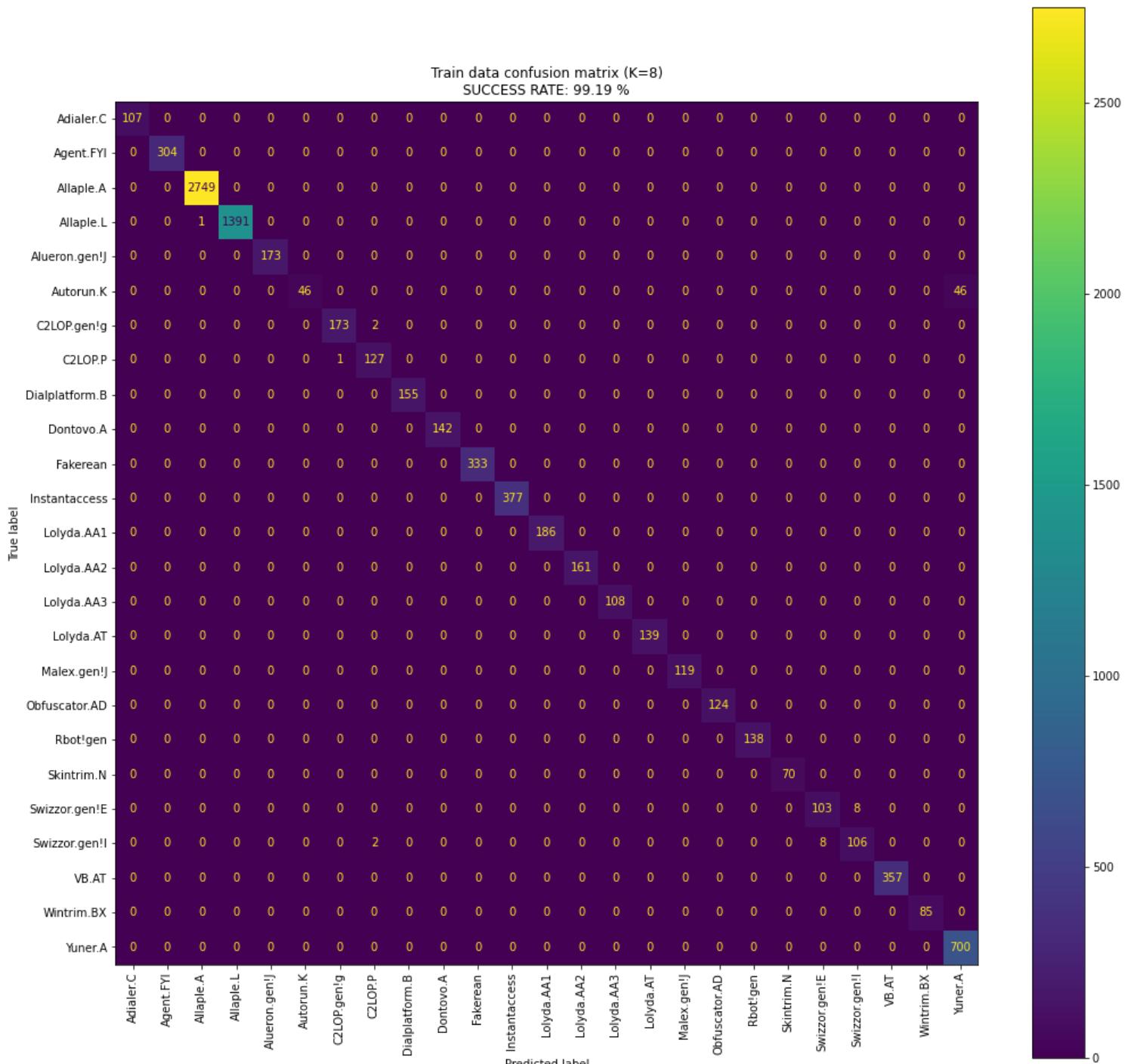
K = 8

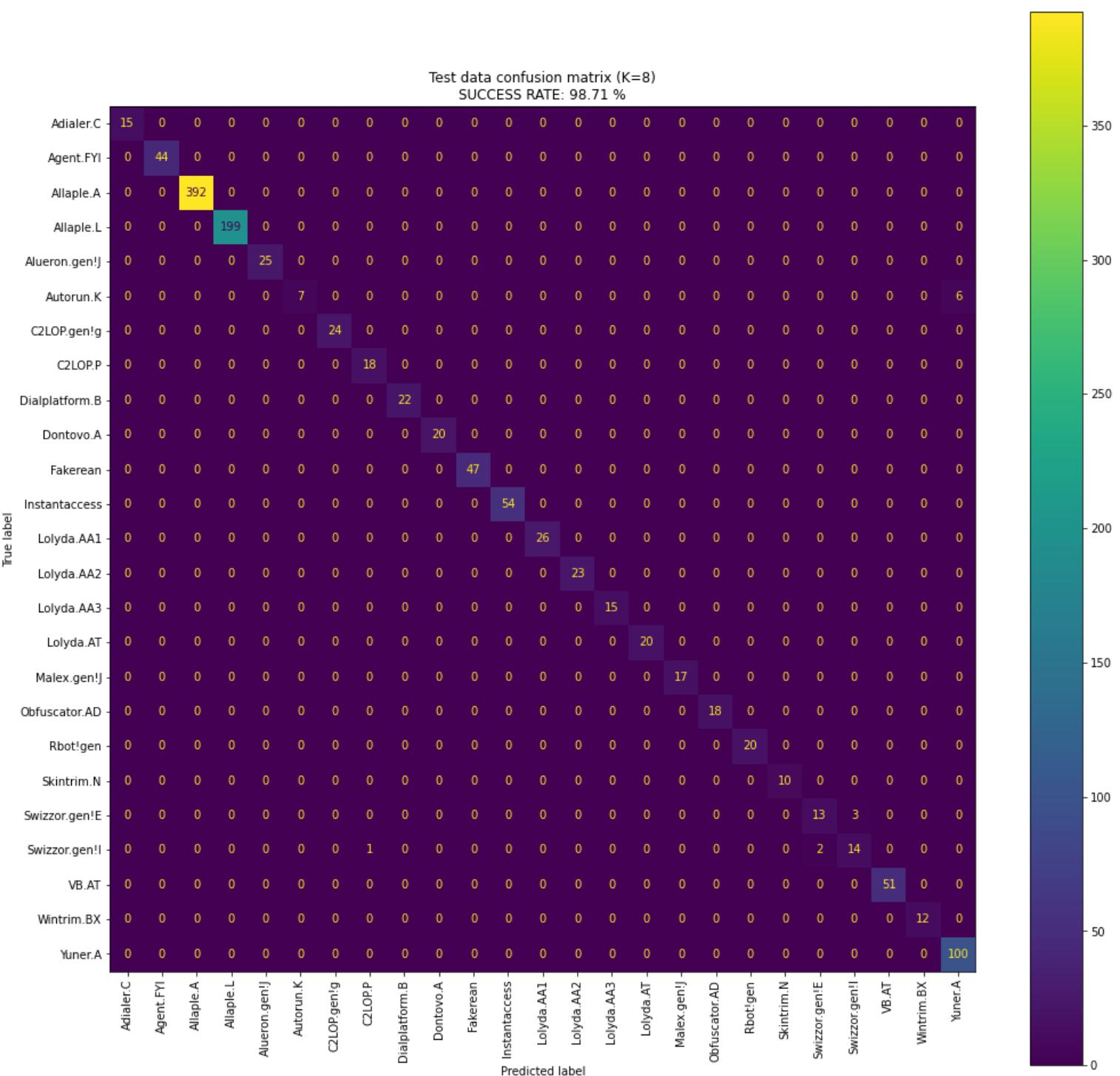
Success Rate (after normalizing):

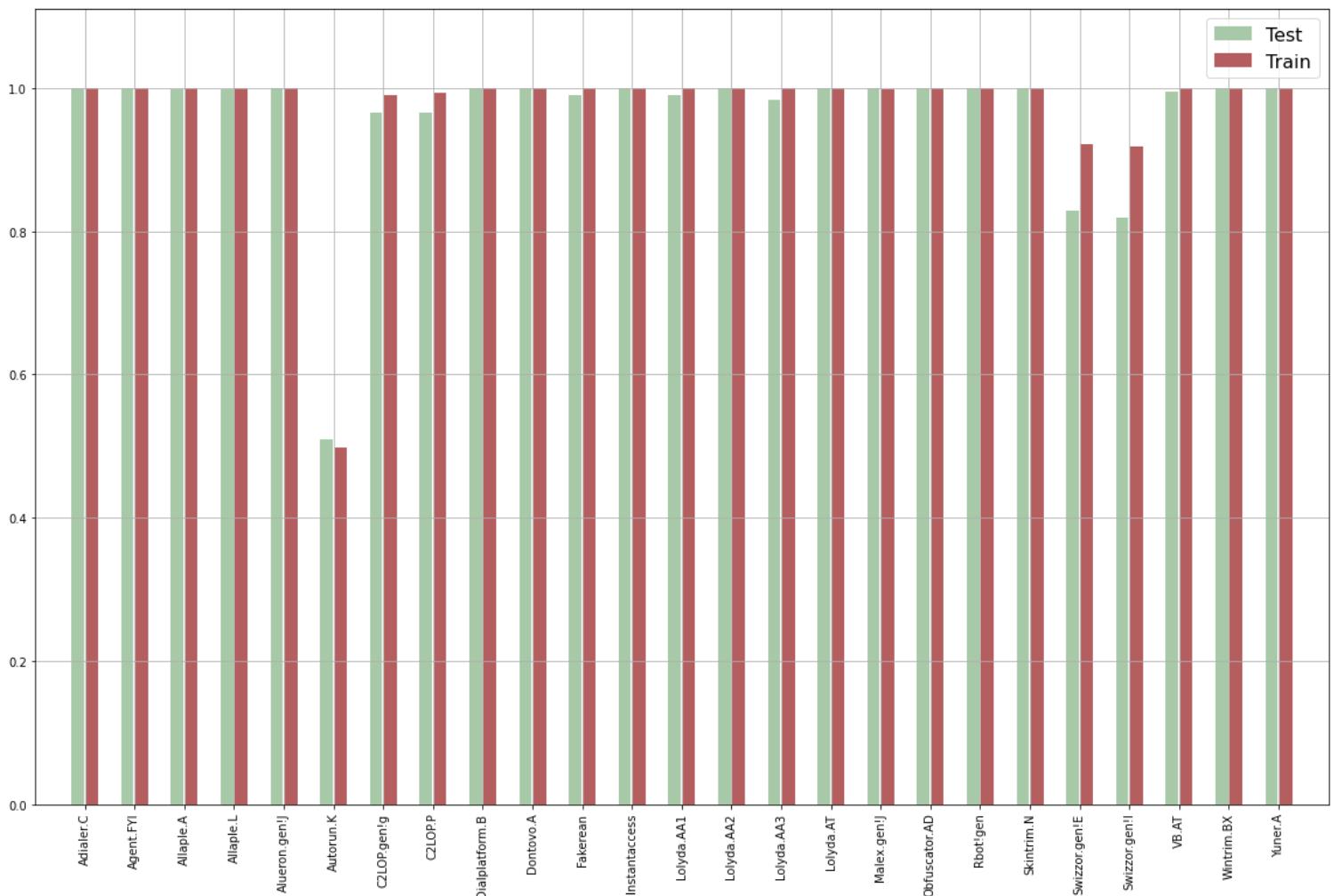
- Train: 99.188 %
- Test: 98.710 %

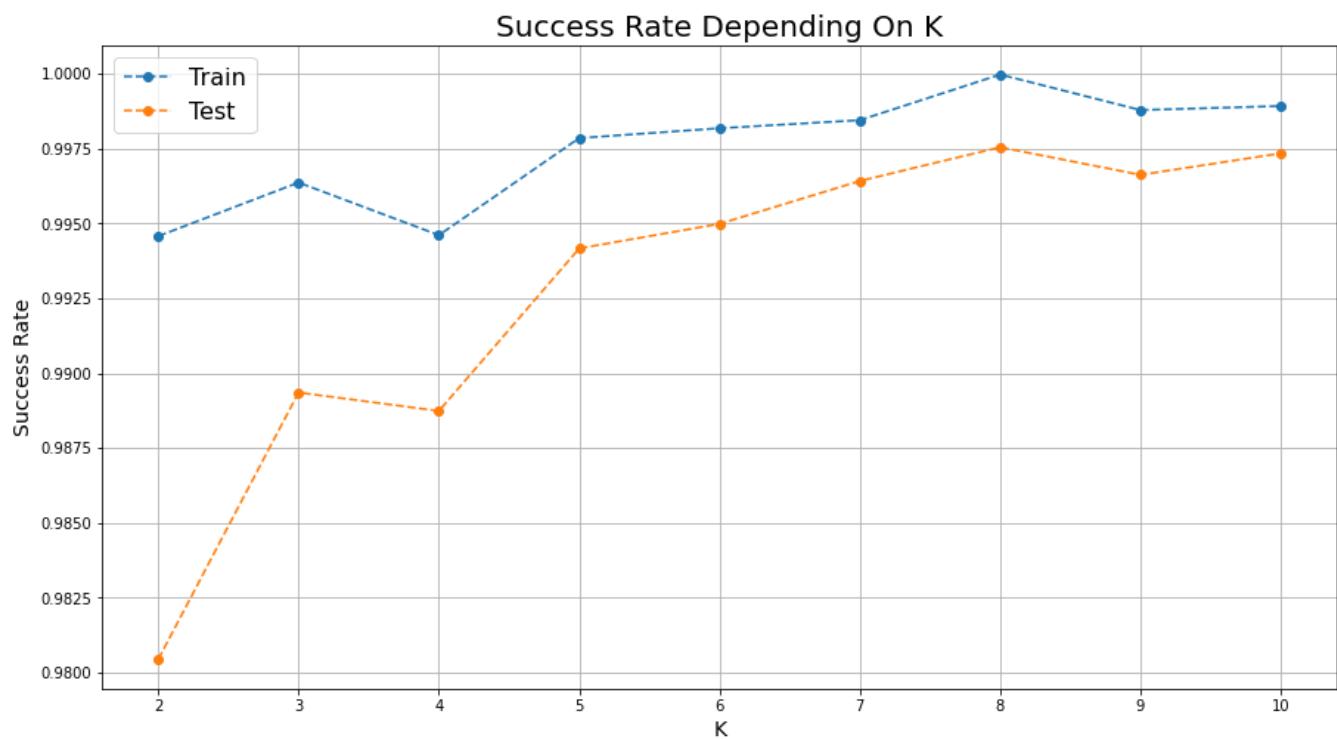
Malware family with the highest false classification (after normalizing):

- Train: Autorun.K
- Test: Autorun.K









Model Summary:

Model: "sequential"

Layer (type)	Output Shape	Param #
=====		
conv2d (Conv2D)	(None, 64, 64, 32)	320
=====		
flatten (Flatten)	(None, 131072)	0
=====		
dense (Dense)	(None, 25)	3276825
=====		
Total params: 3,277,145		
Trainable params: 3,277,145		
Non-trainable params: 0		

Train & Test (Includes test results)

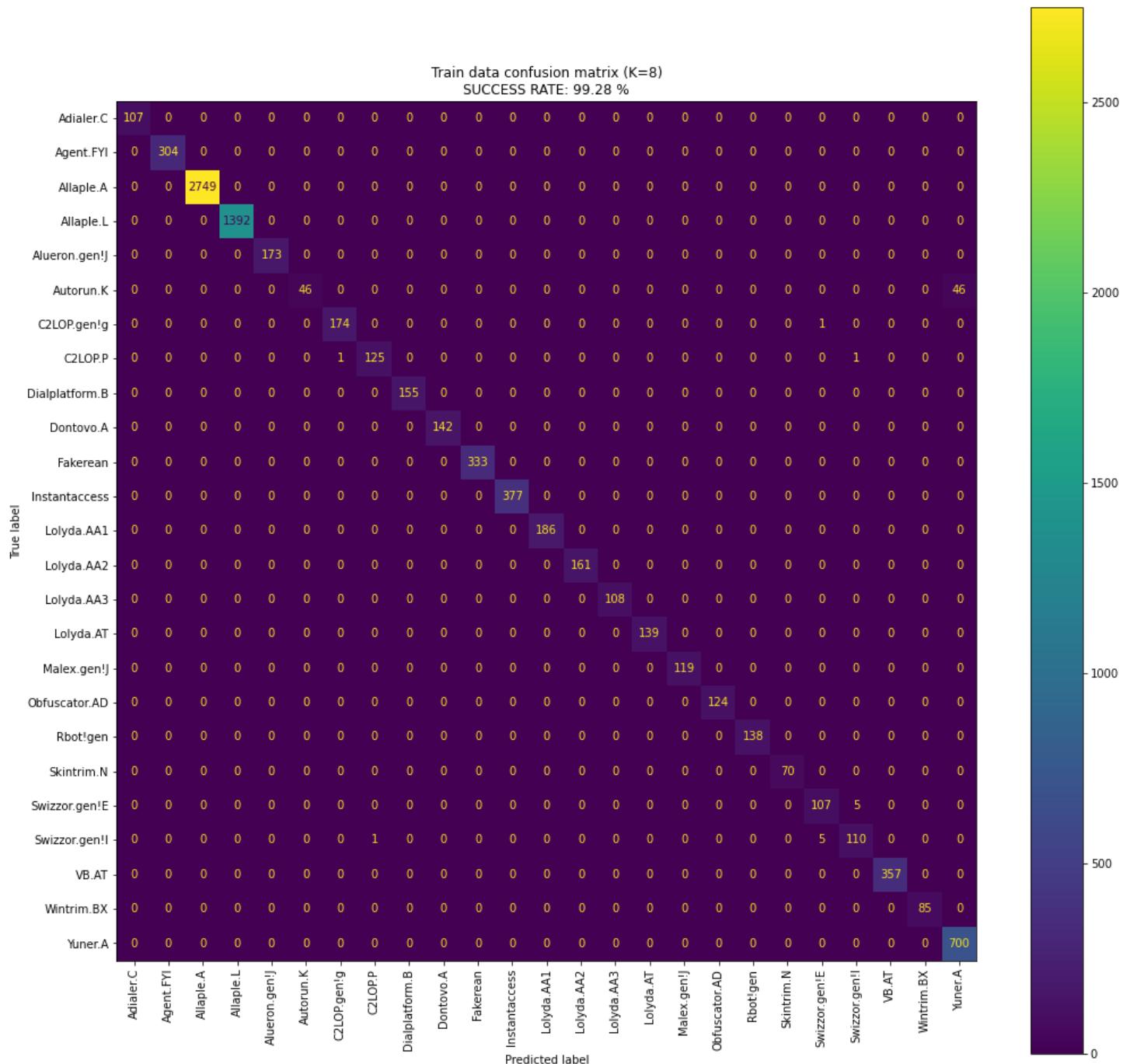
K = 8

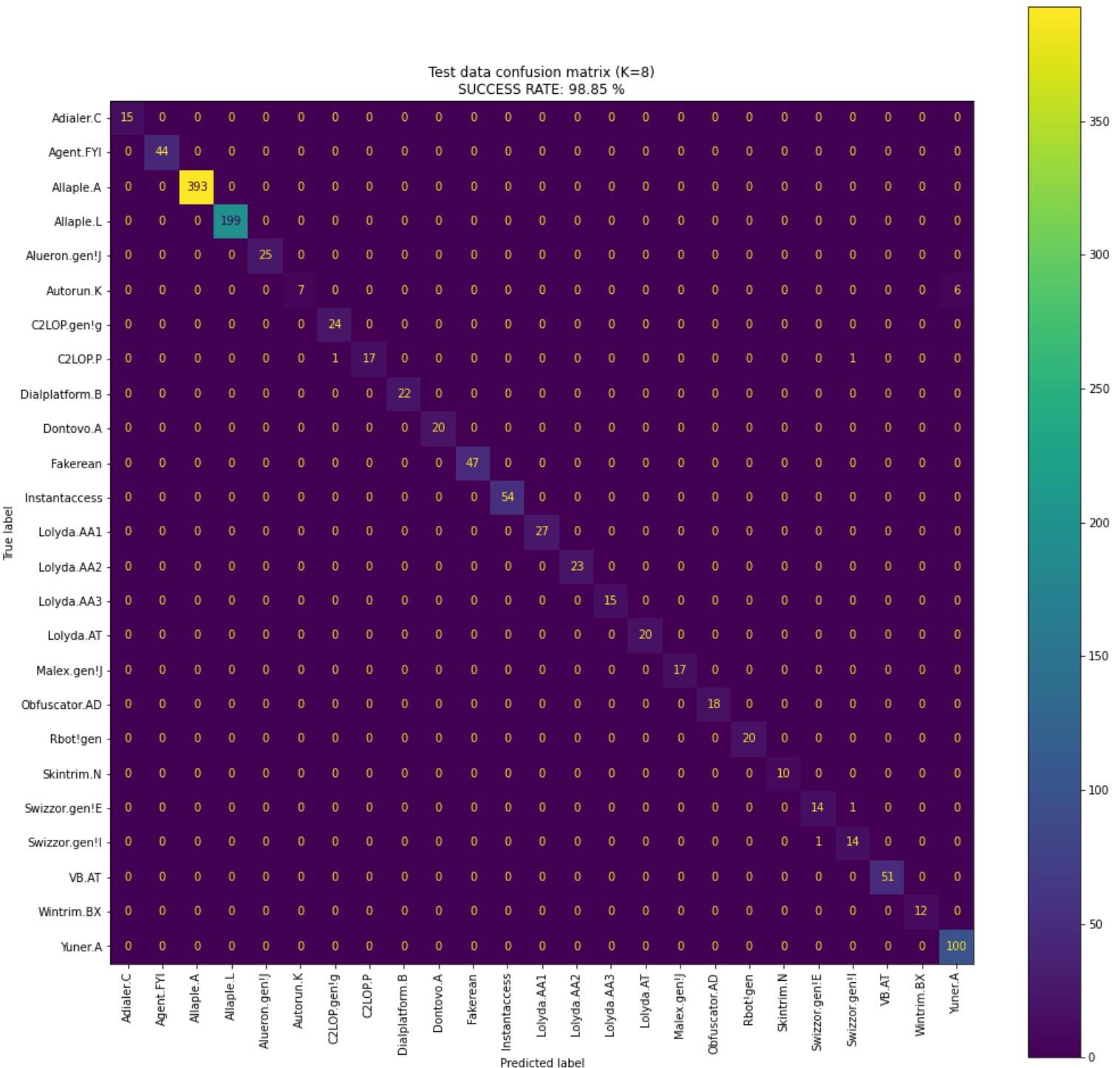
Success Rate (after normalizing):

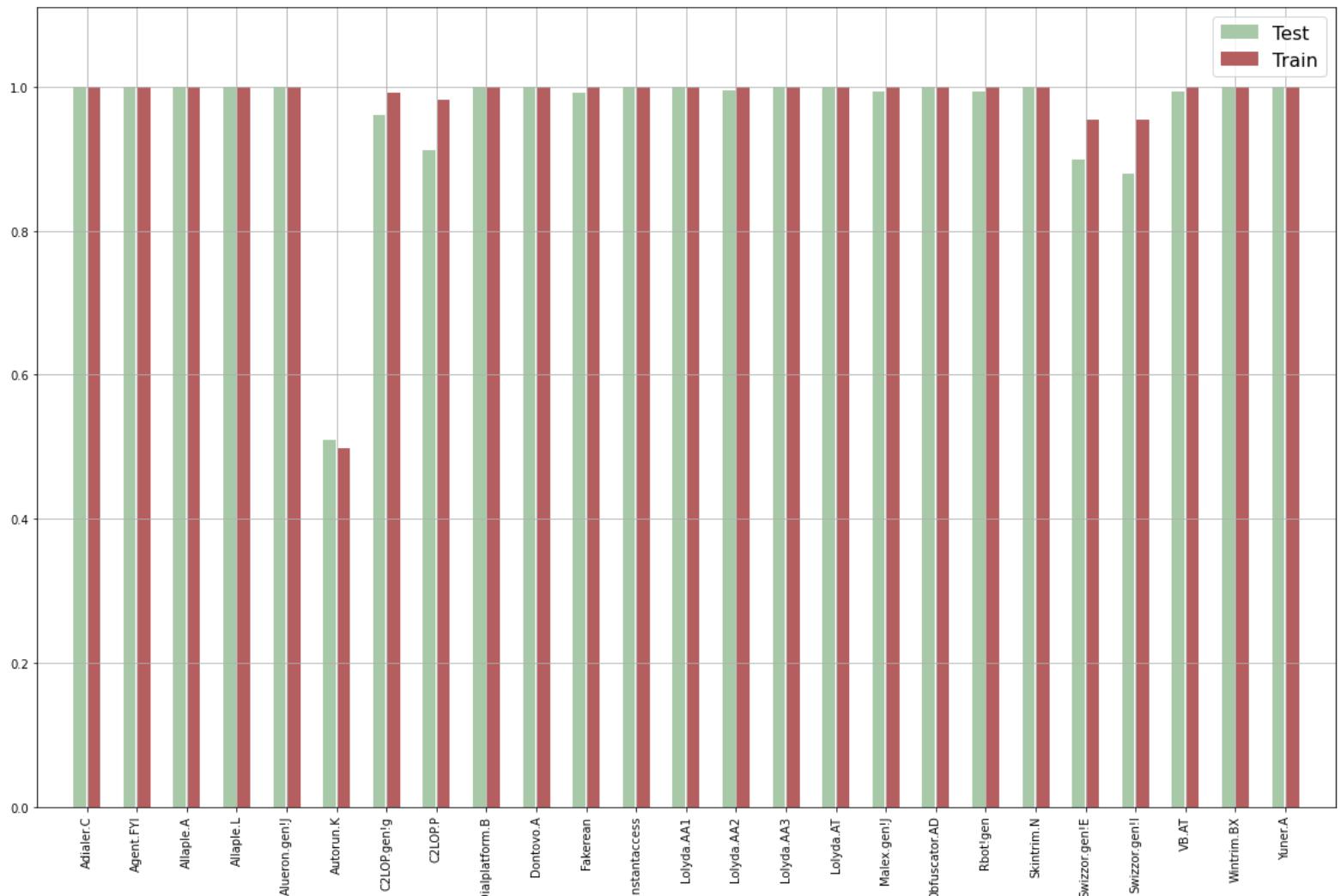
- Train: 99.997 %
- Test: 99.754 %
- AVG: 99.967 %

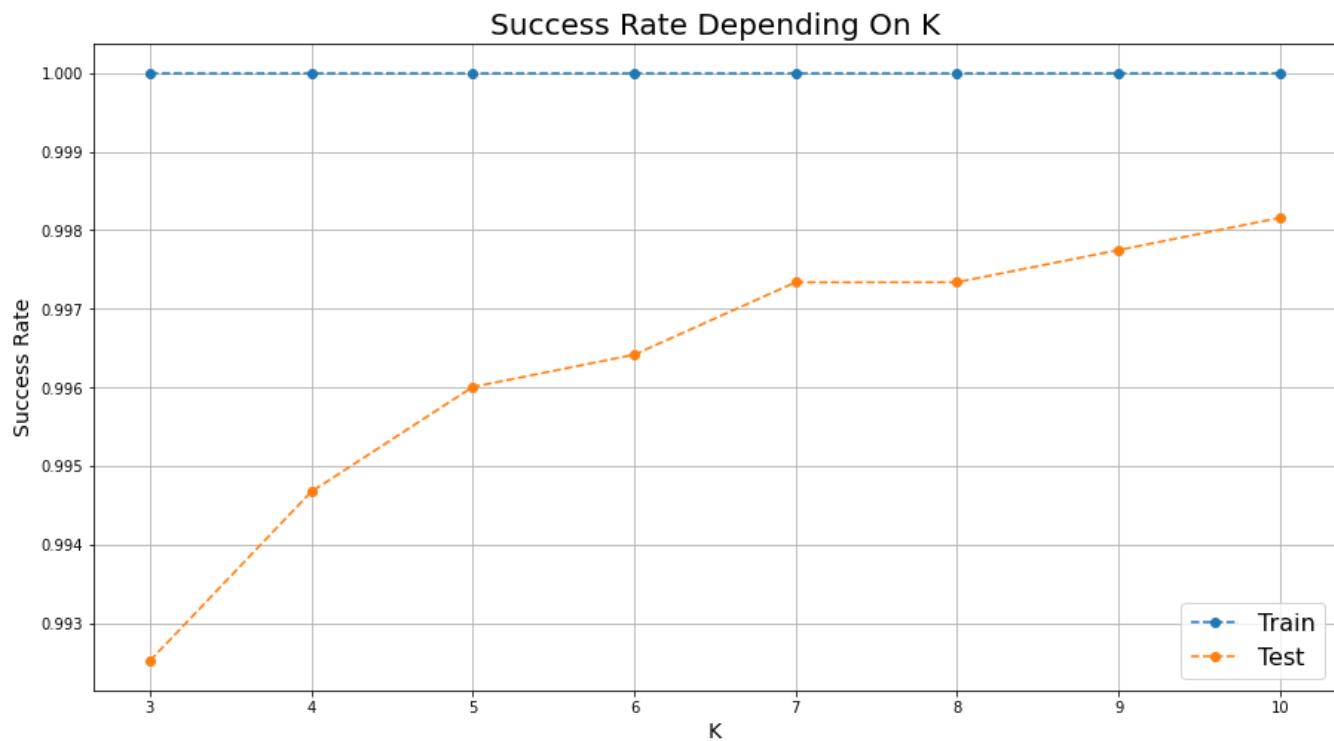
Malware family with the highest false classification (after normalizing):

- Train: Swizzor.gen!I
- Test: Swizzor.gen!I









Model Summary:

Model: "sequential"

Layer (type)	Output Shape	Param #
=====		
conv2d (Conv2D)	(None, 128, 128, 32)	320
flatten (Flatten)	(None, 524288)	0
dense (Dense)	(None, 25)	13107225
=====		
Total params:	13,107,545	
Trainable params:	13,107,545	
Non-trainable params:	0	

Train & Test (Includes test results)

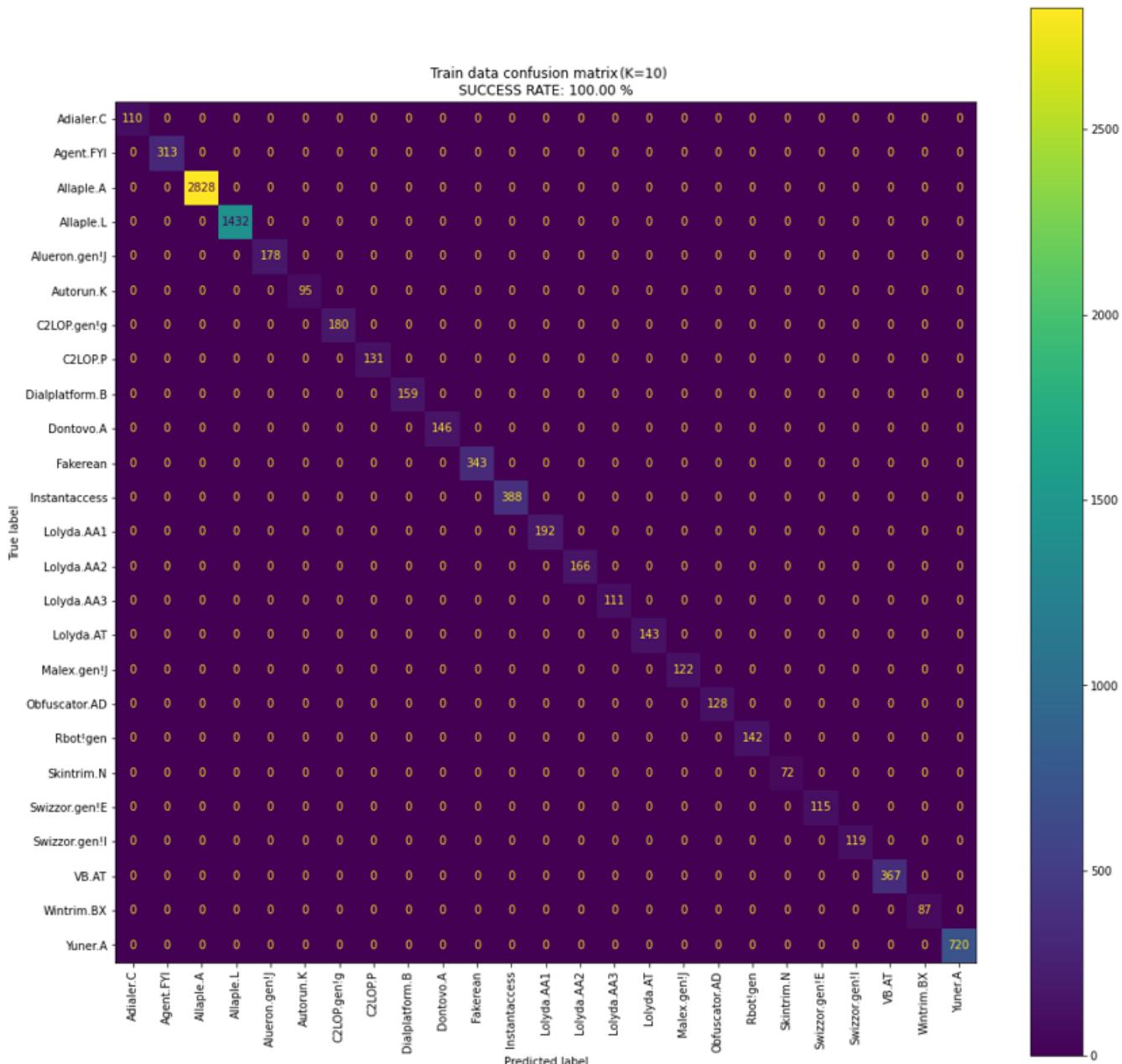
K = 10

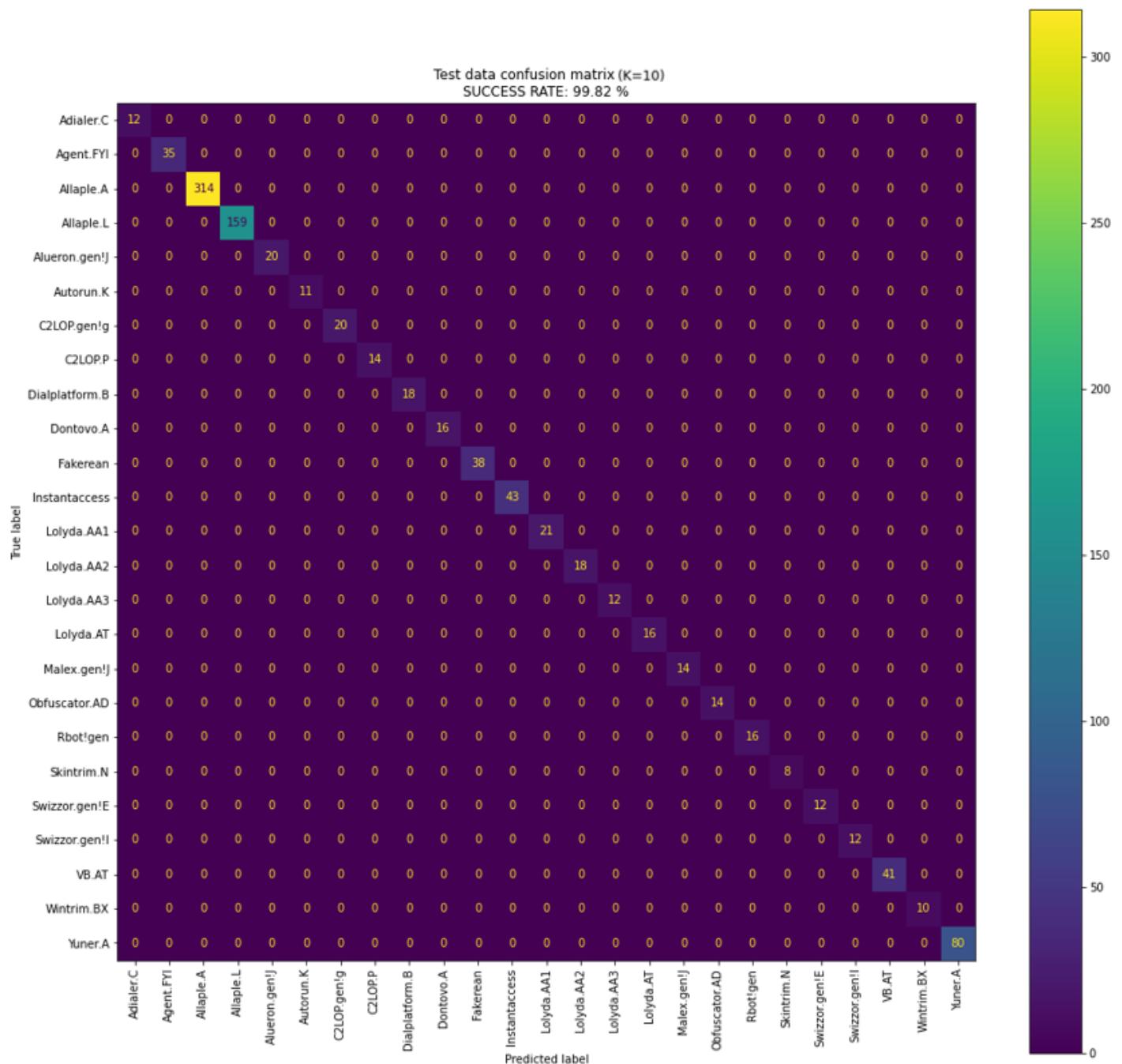
Success Rate (after normalizing):

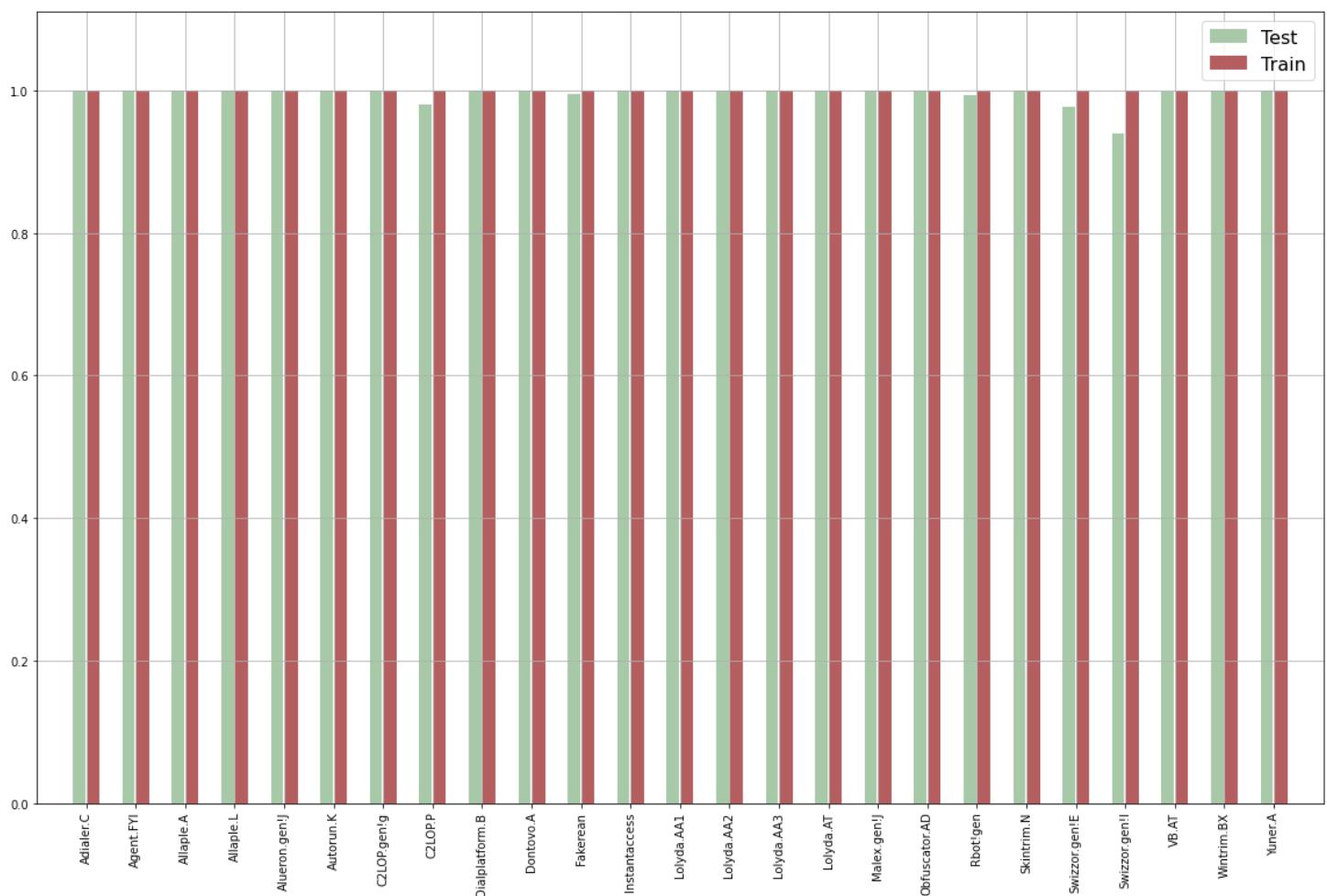
- Train: 100.000 %
- Test: 99.816 %

Malware family with the highest false classification (after normalizing):

- Train: N/A
- Test: Swizzor.gen!I





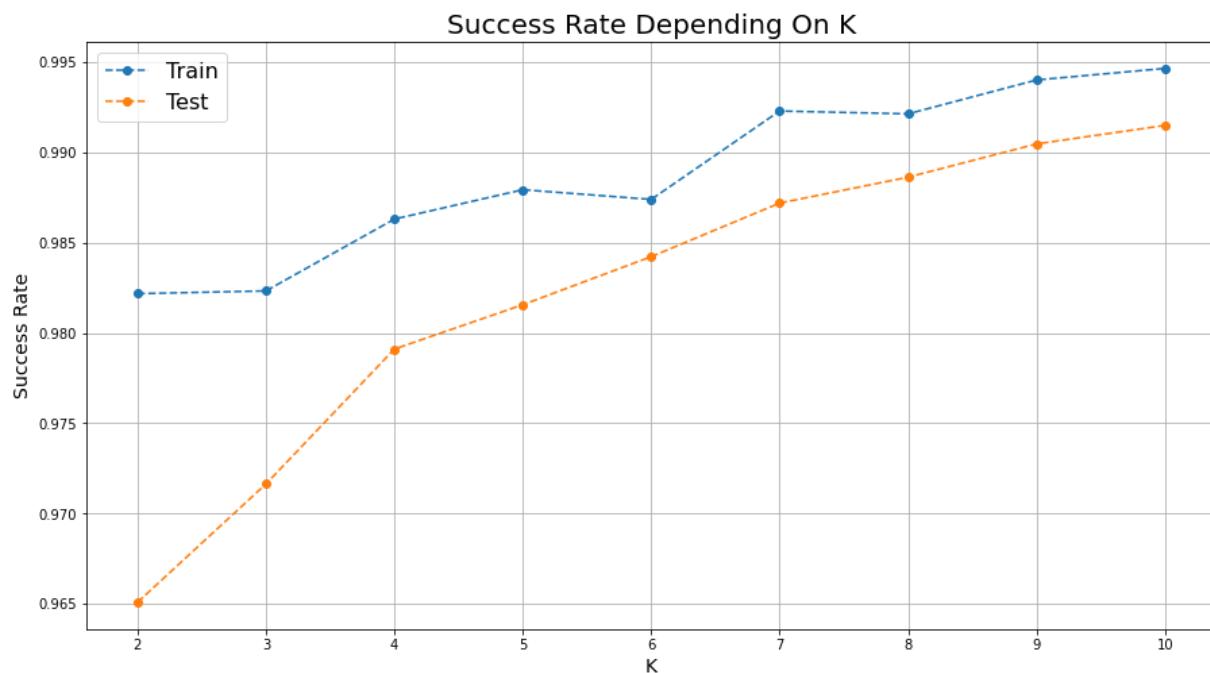


מודל CNN שכבה 1, 64 פילטרים 5.4.2

CNN (1 Layer, 64 Filter, 11 Epochs)				
Size	Best K - Test	Test Accuracy	Best K - Train	Train Accuracy
32x32	10	99.15	10	99.466
64x64	10	99.826	10	100
128x128	9	99.775	9	100

טבלה 5: תוצאות CNN שכבה 1, 64 פילטרים

32×32 5.4.2.1



Model Summary:

Model: "sequential"

Layer (type)	Output Shape	Param #
<hr/>		
conv2d (Conv2D)	(None, 32, 32, 64)	640
flatten (Flatten)	(None, 65536)	0
dense (Dense)	(None, 25)	1638425
<hr/>		
Total params:	1,639,065	
Trainable params:	1,639,065	
Non-trainable params:	0	

Train & Test (Includes test results)

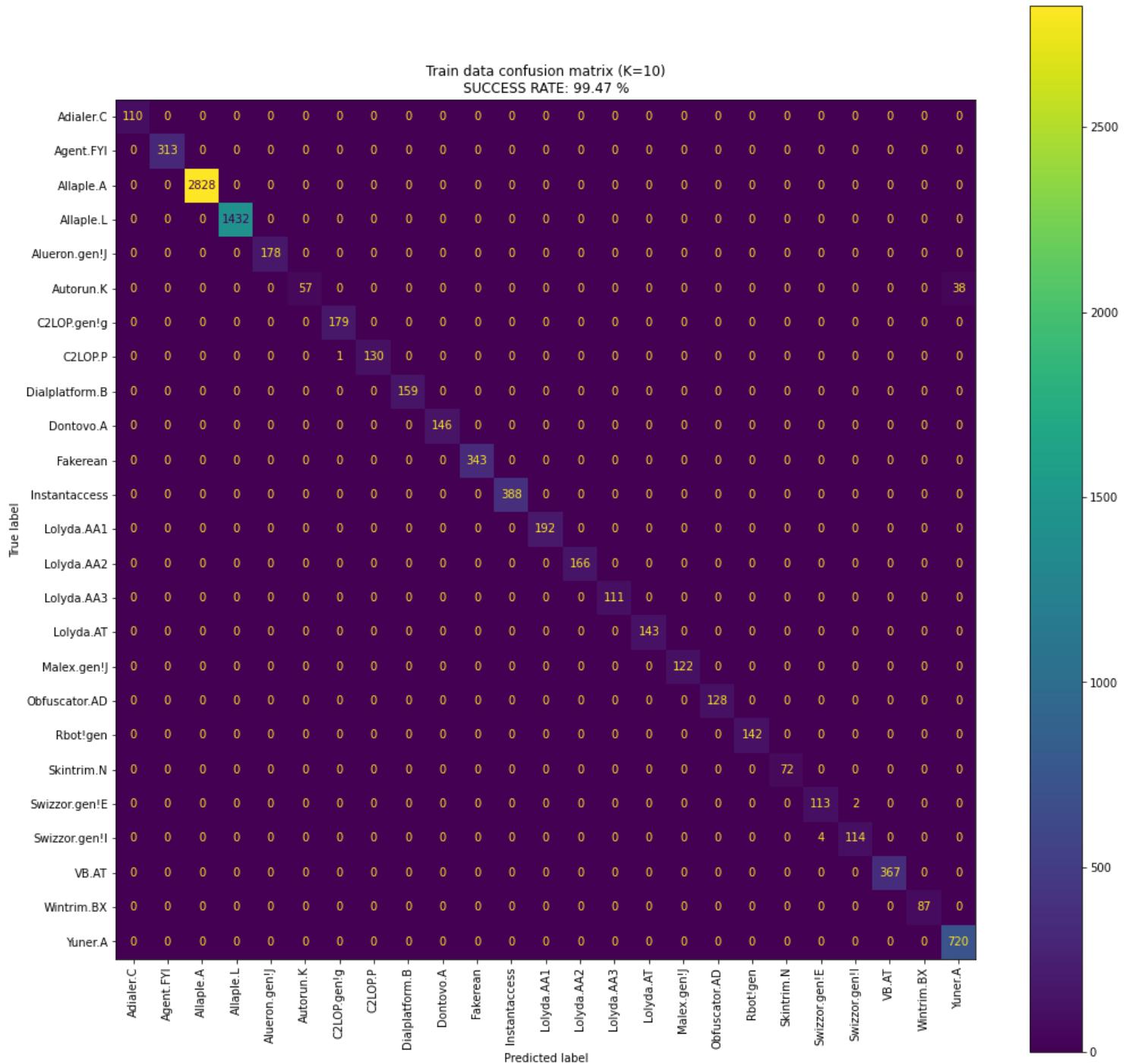
K = 10

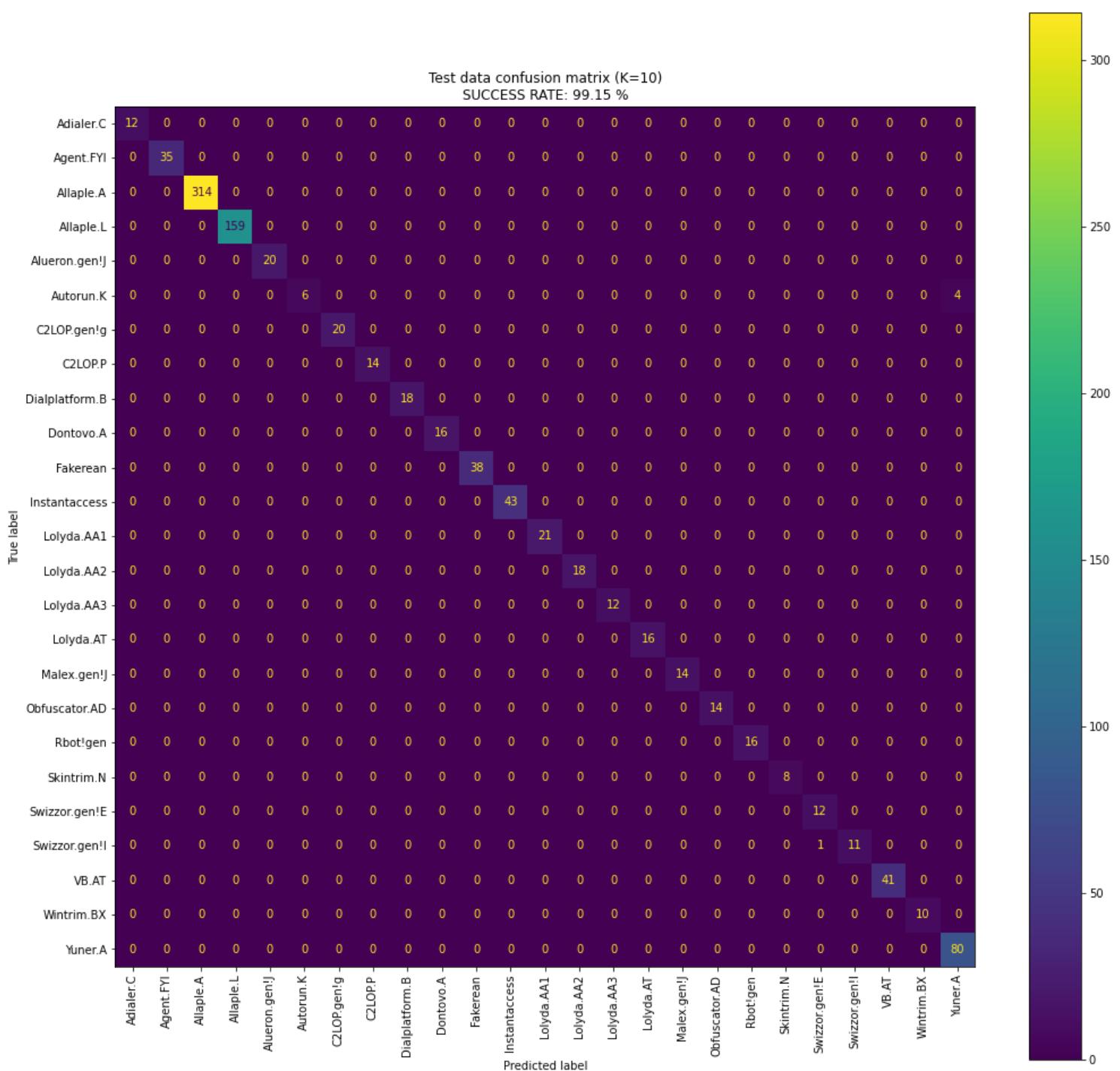
Success Rate (after normalizing):

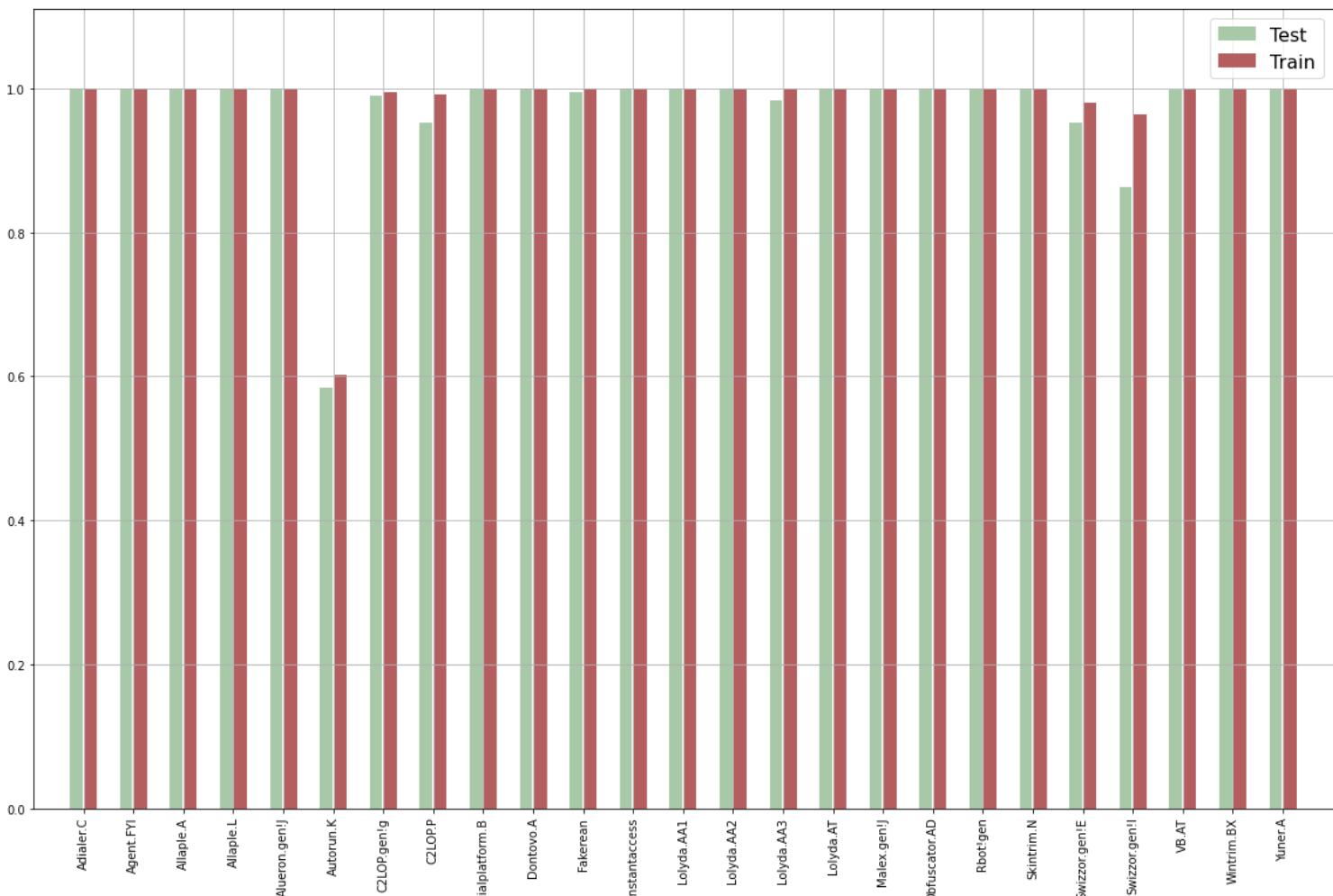
- Train: 99.466 %
- Test: 99.150 %

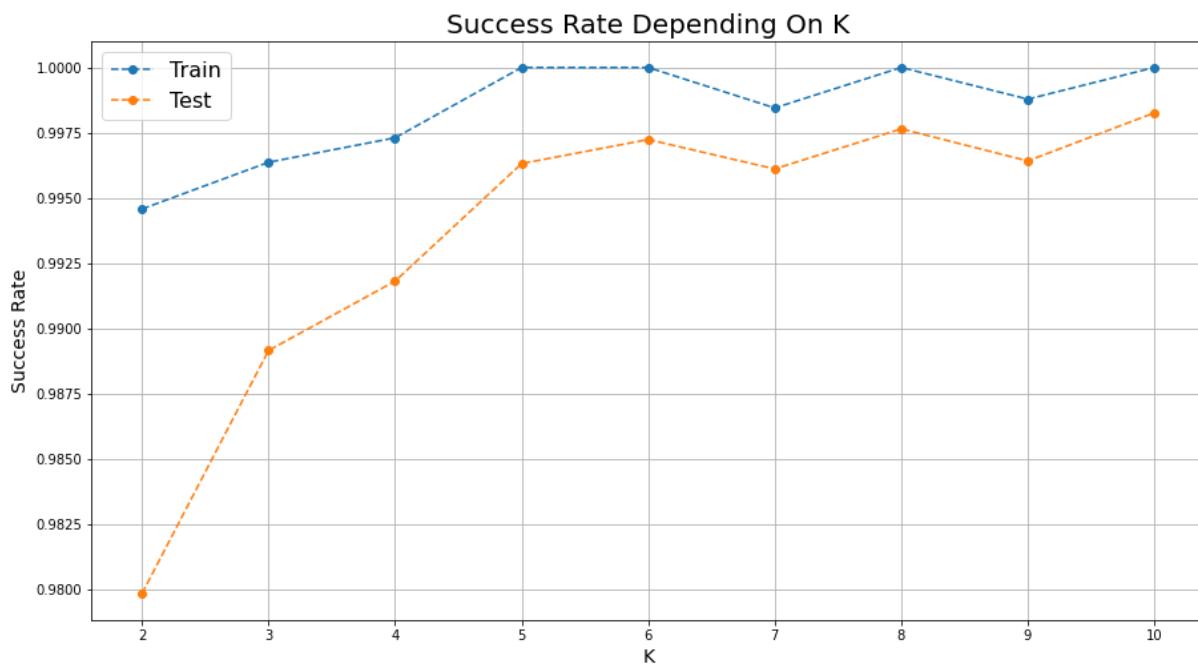
Malware family with the highest false classification (after normalizing):

- Train: Autorun.K
- Test: Autorun.K









Model Summary:

Model: "sequential"

Layer (type)	Output Shape	Param #
<hr/>		
conv2d (Conv2D)	(None, 64, 64, 64)	640
flatten (Flatten)	(None, 262144)	0
dense (Dense)	(None, 25)	6553625
<hr/>		
Total params: 6,554,265		
Trainable params: 6,554,265		
Non-trainable params: 0		

Train & Test (Includes test results)

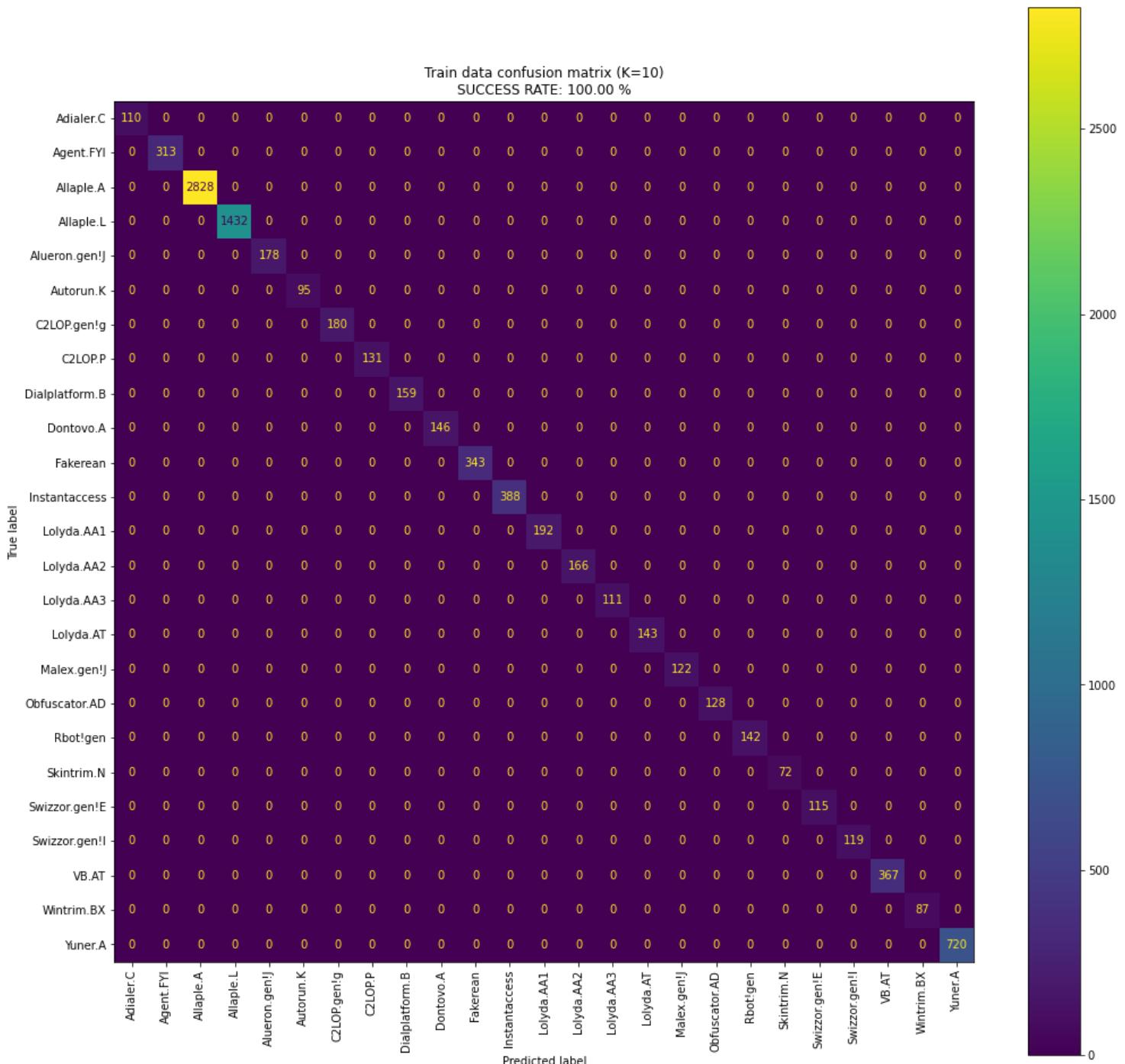
$$K = 10$$

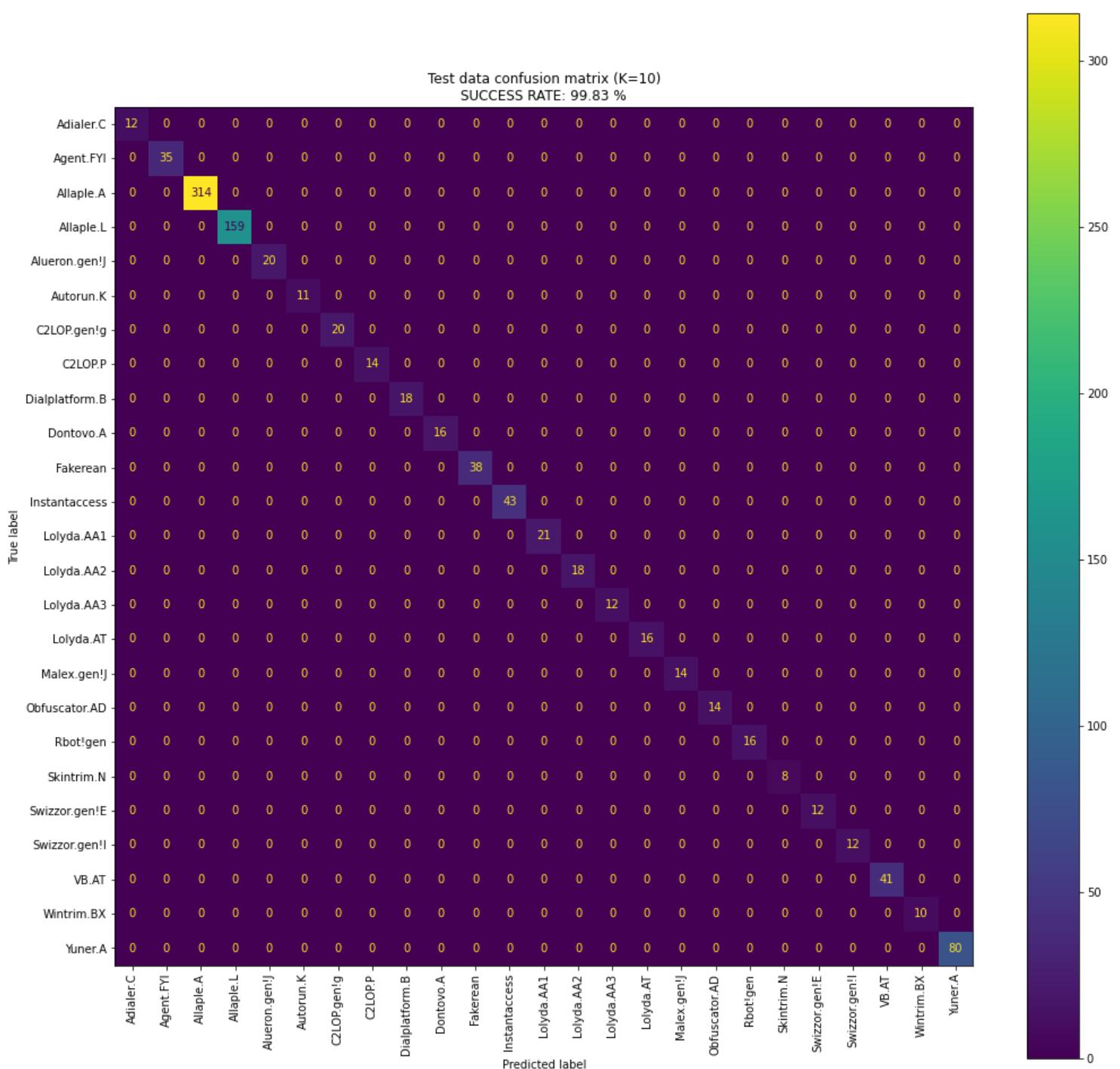
Success Rate (after normalizing):

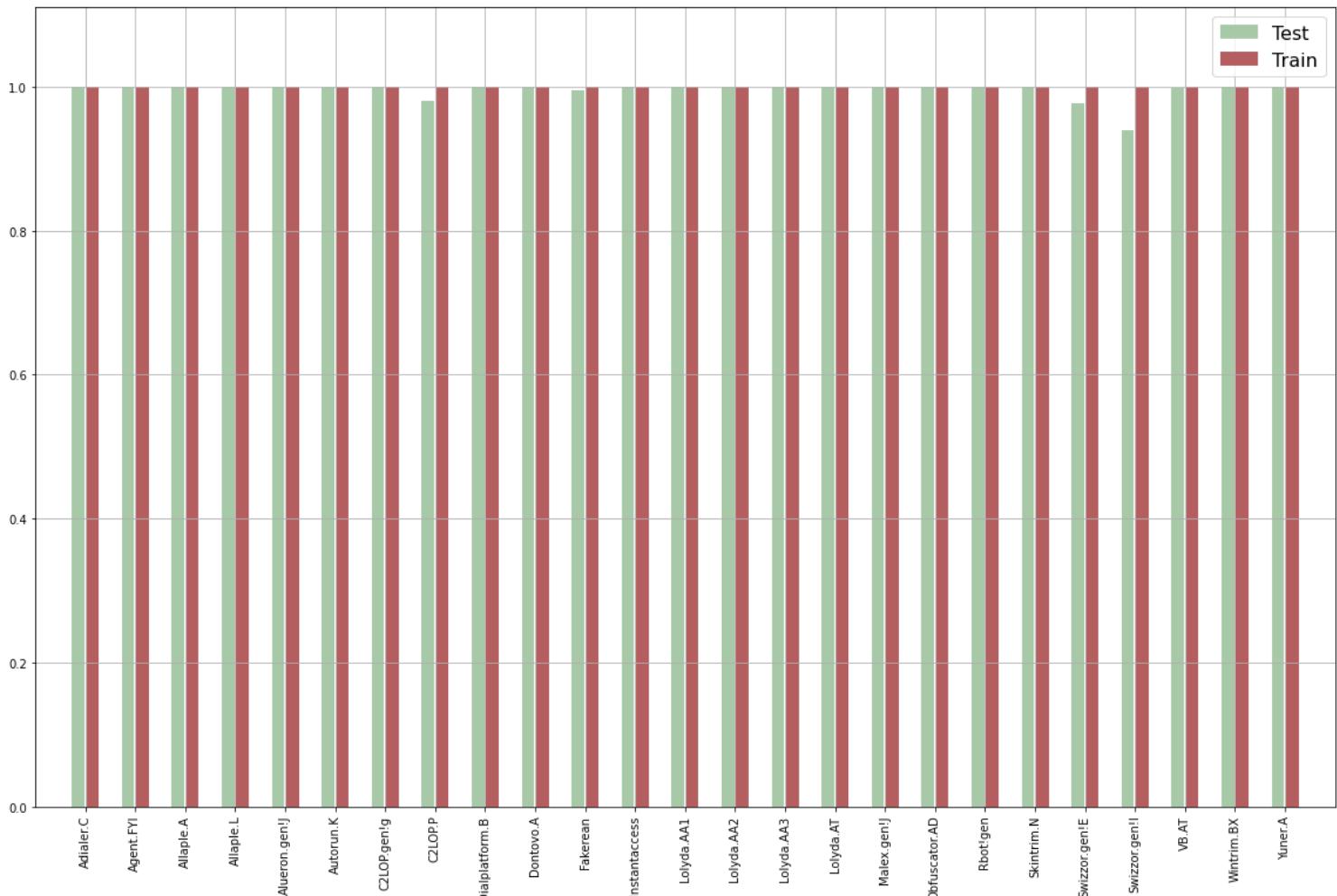
- Train: 100.000 %
 - Test: 99.826 %

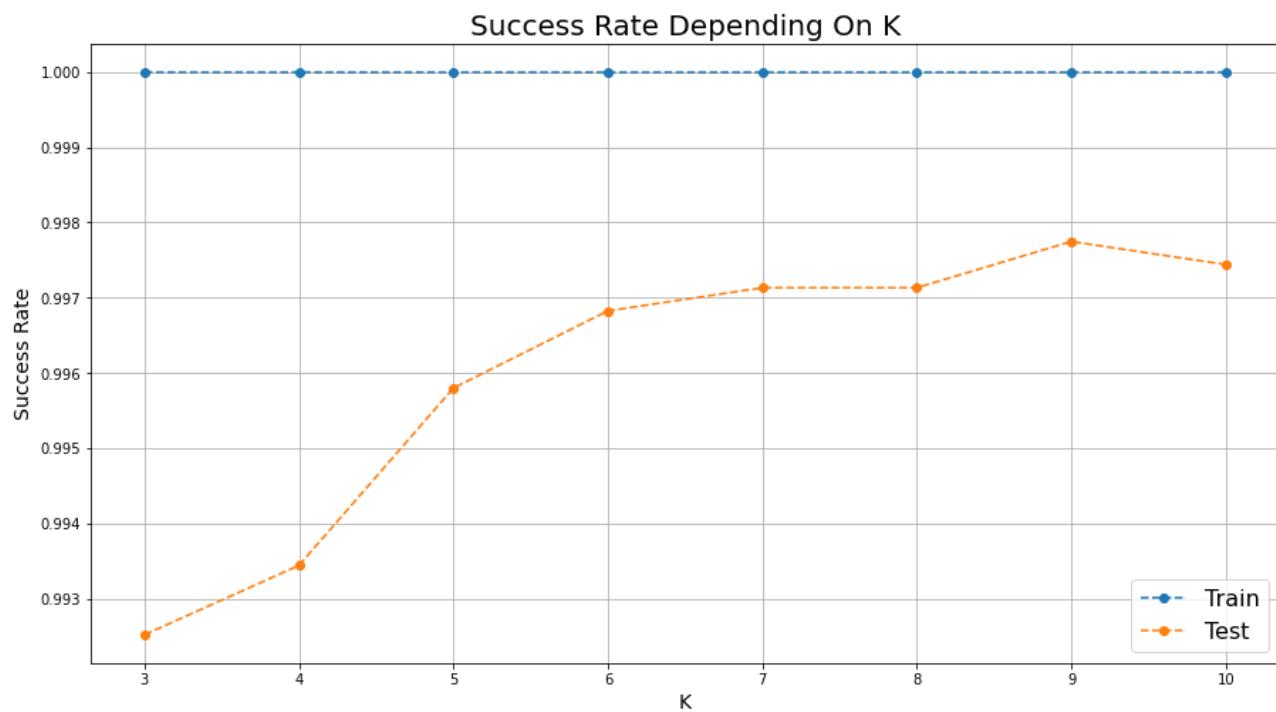
Malware family with the highest false classification (after normalizing):

- Train: N/A
 - Test: Swizzor.gen!I









Model Summary:

Model: "sequential"

Layer (type)	Output Shape	Param #
<hr/>		
conv2d (Conv2D)	(None, 128, 128, 64)	640
<hr/>		
flatten (Flatten)	(None, 1048576)	0
<hr/>		
dense (Dense)	(None, 25)	26214425
<hr/>		
Total params: 26,215,065		
Trainable params: 26,215,065		
Non-trainable params: 0		

Train & Test (Includes test results)

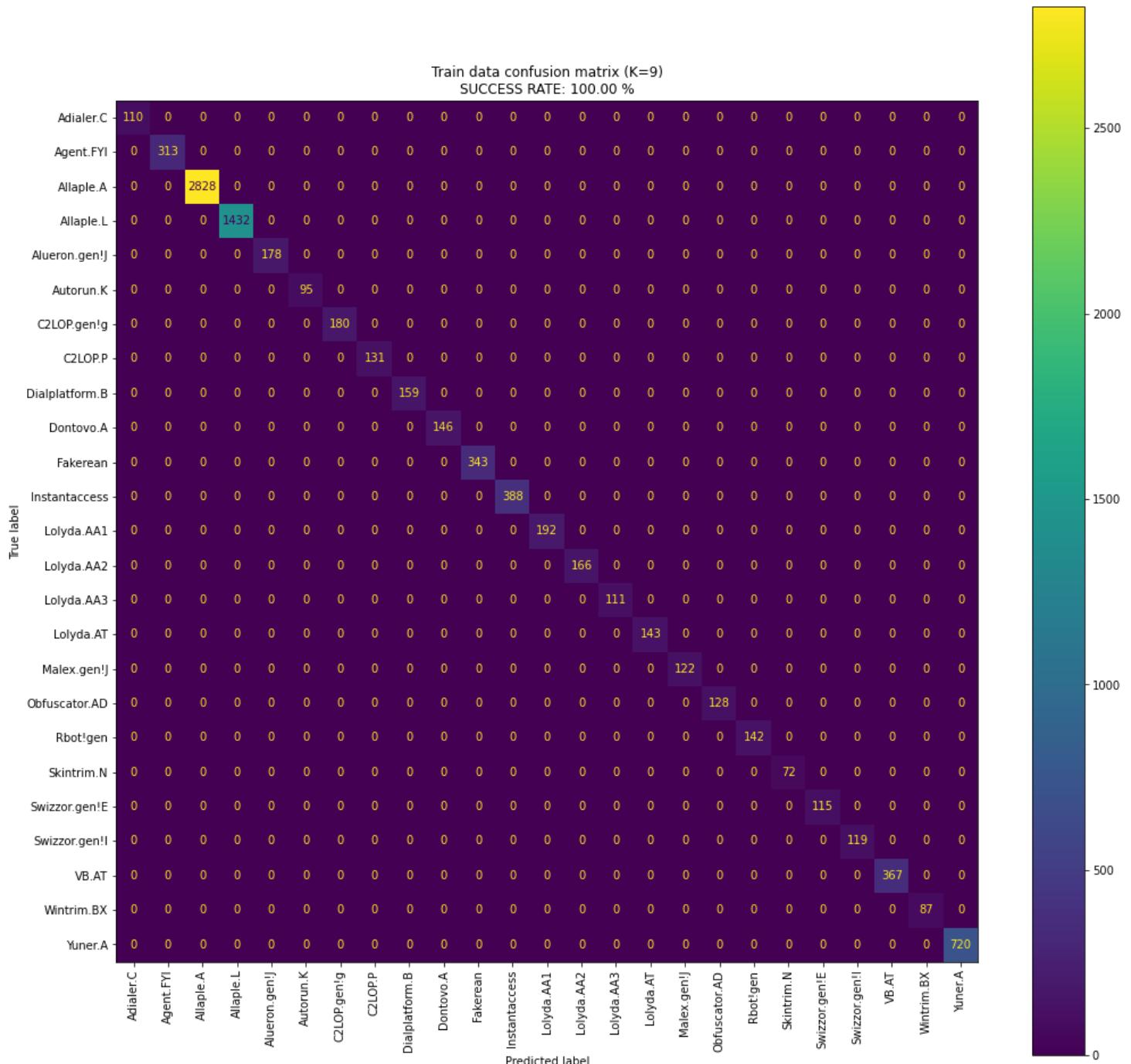
K = 9

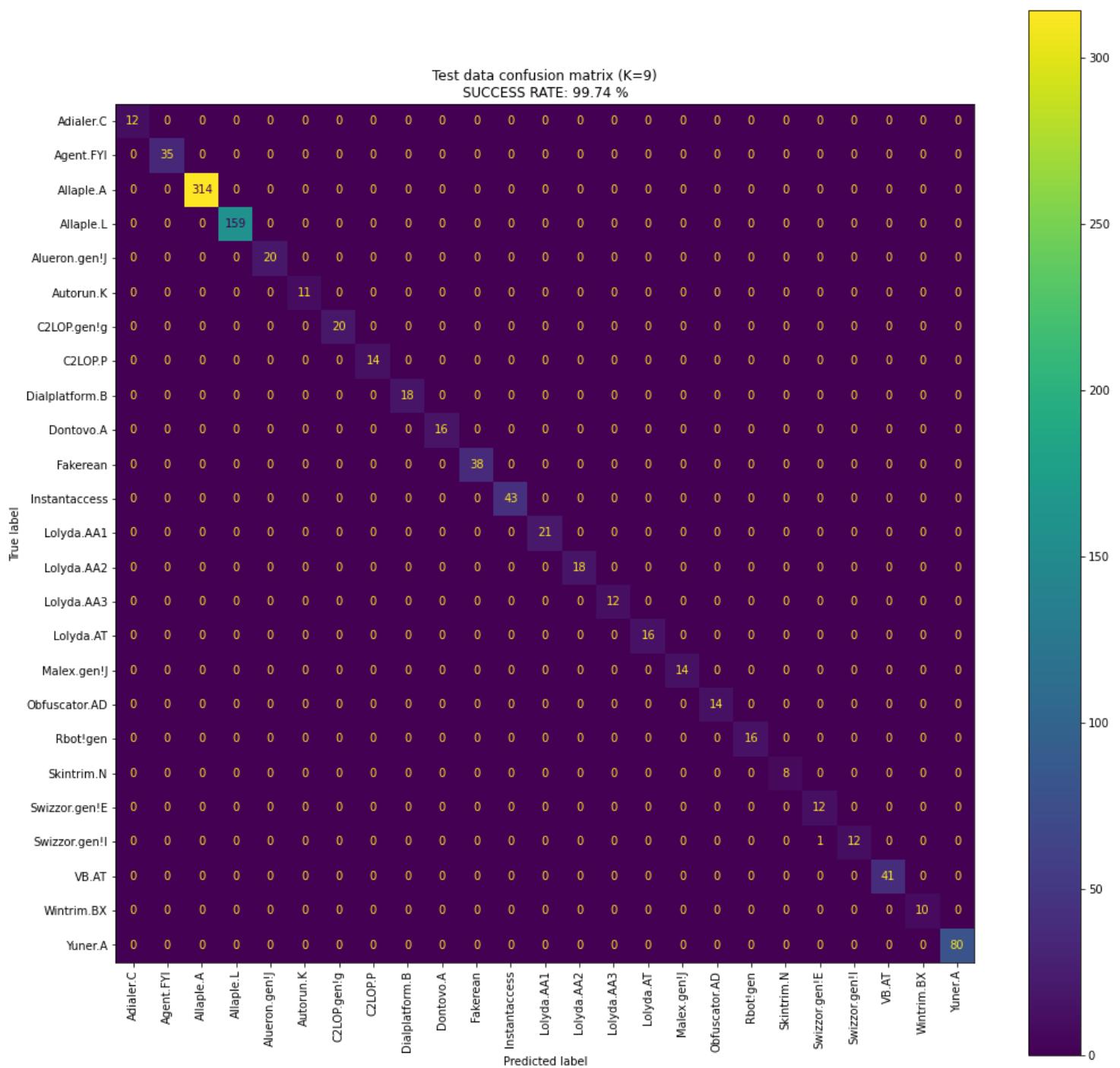
Success Rate (after normalizing):

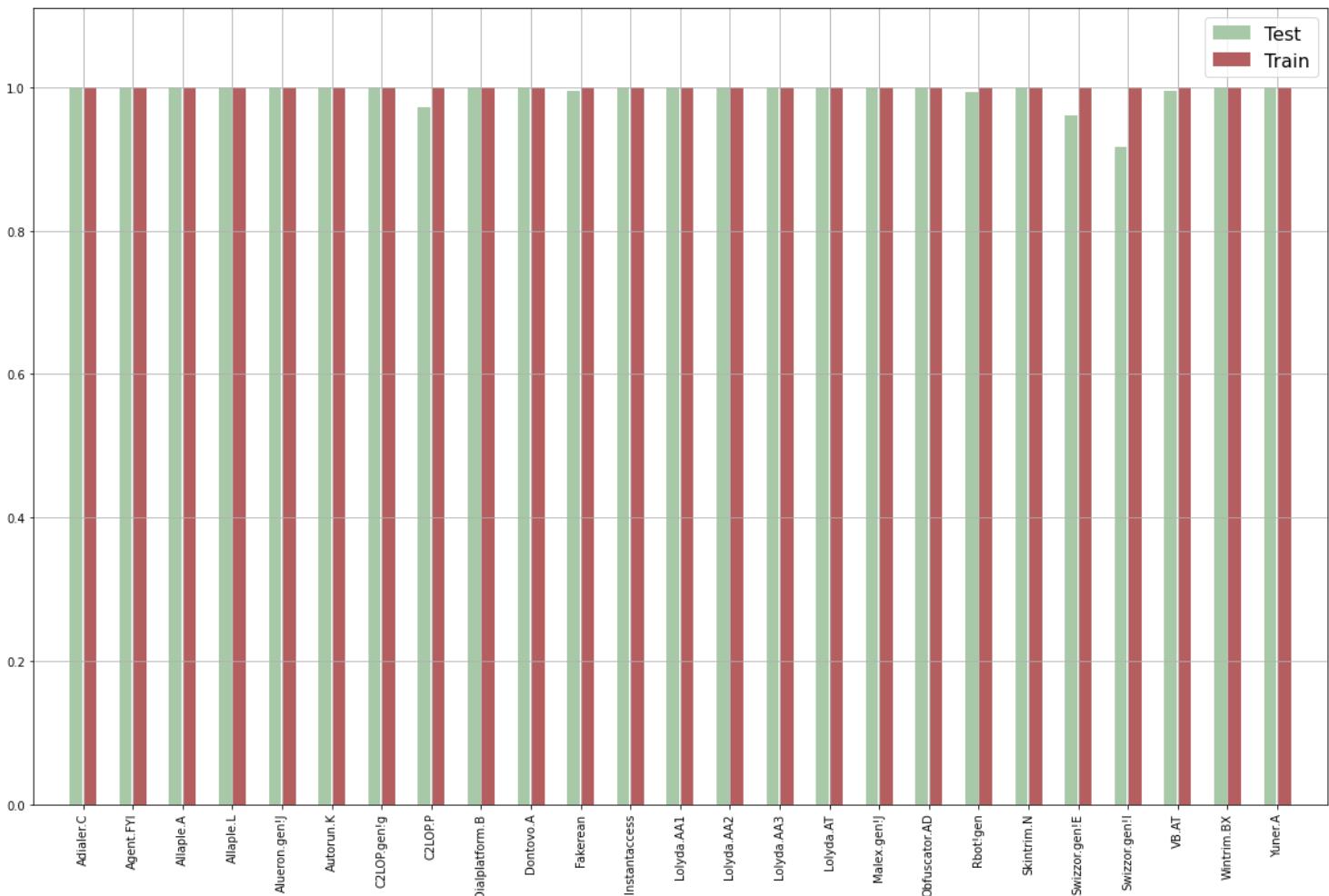
- Train: 100.000 %
- Test: 99.744 %

Malware family with the highest false classification (after normalizing):

- Train: N/A
- Test: Swizzor.gen!I





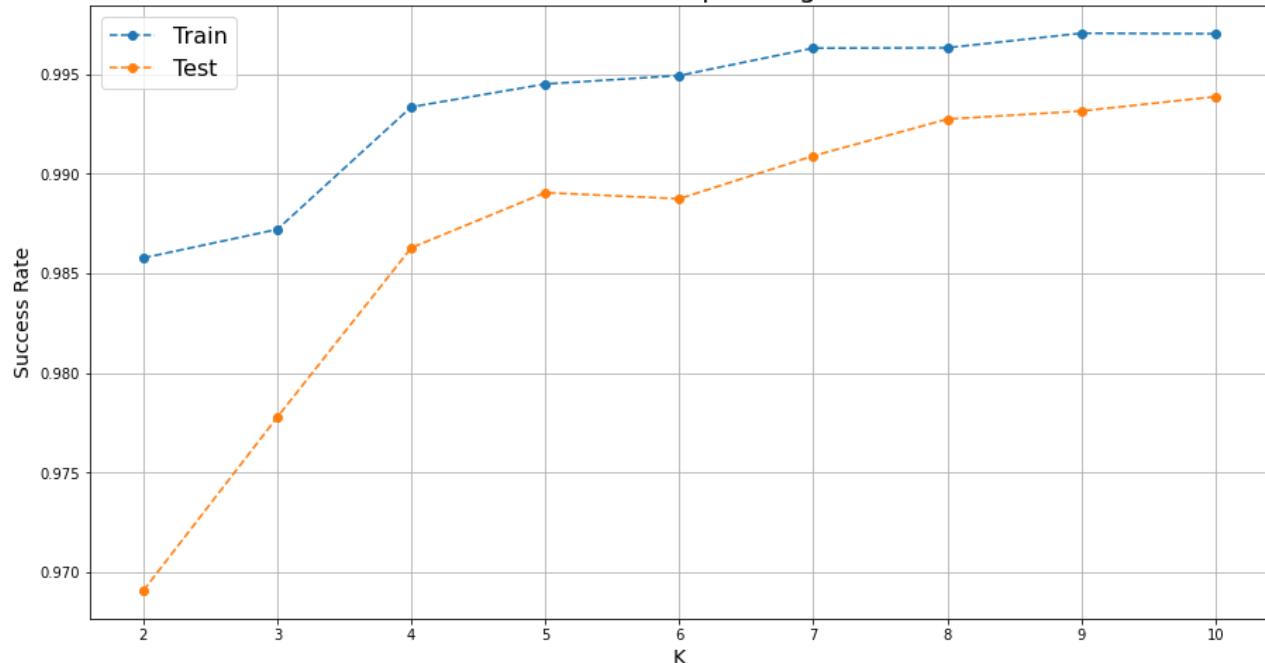


CNN (2 Layers, 32 Filter, 11 Epochs)				
Size	Best K - Test	Test Accuracy	Best K - Train	Train Accuracy
32x32	10	99.386	10	99.702
64x64	9	99.724	9	100
128x128	10	99.775	10	100

טבלה 6: תוצאות CNN 2 שכבות, 32 פילטרים

32 × 32 5.4.3.1

Success Rate Depending On K



Model Summary:

Model: "sequential"

Layer (type)	Output Shape	Param #
<hr/>		
conv2d (Conv2D)	(None, 32, 32, 32)	320
conv2d_1 (Conv2D)	(None, 30, 30, 32)	9248
flatten (Flatten)	(None, 28800)	0
dense (Dense)	(None, 25)	720025
<hr/>		
Total params: 729,593		
Trainable params: 729,593		
Non-trainable params: 0		

Train & Test (Includes test results)

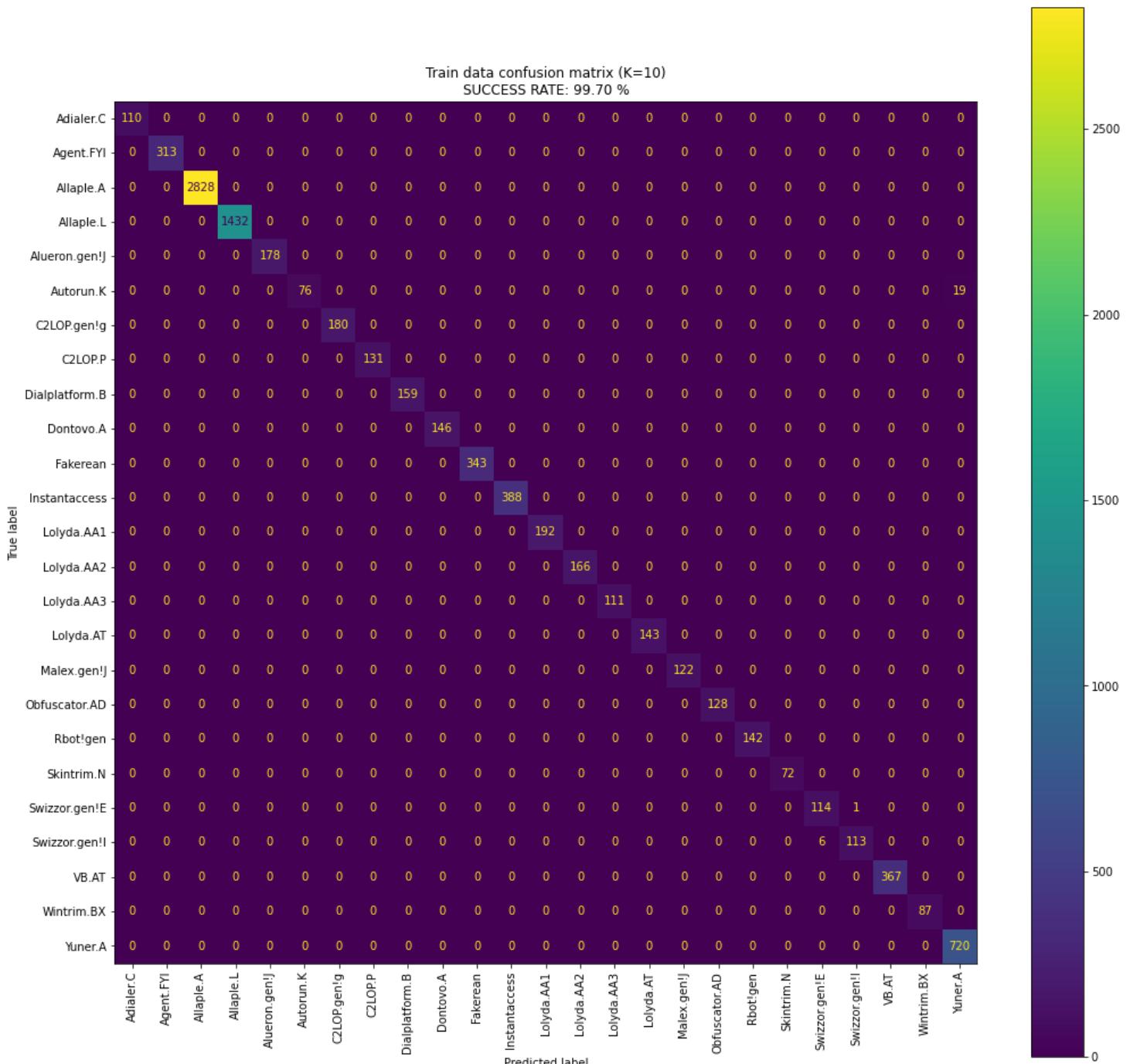
K = 10

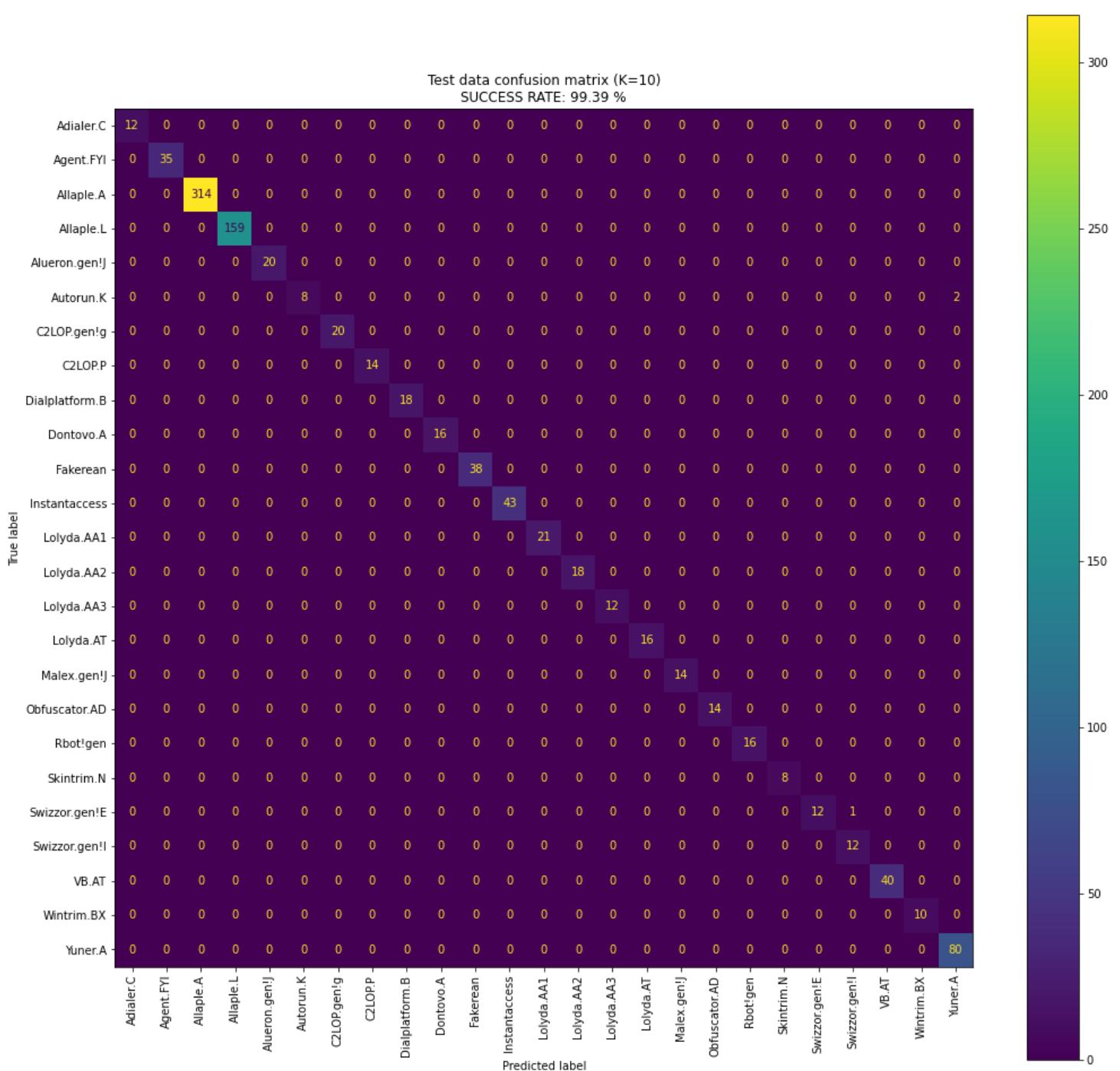
Success Rate (after normalizing):

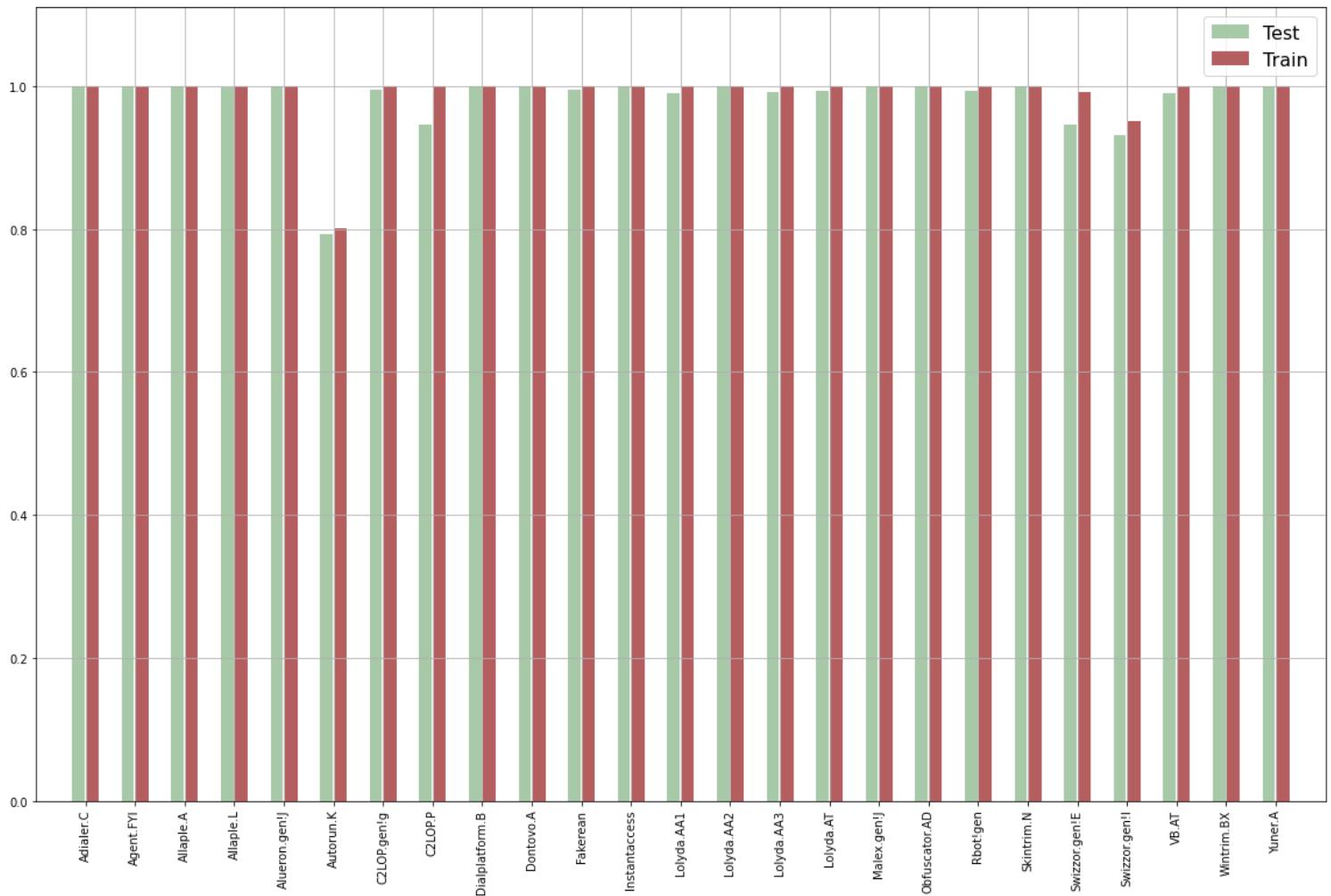
- Train: 99.702 %
- Test: 99.386 %

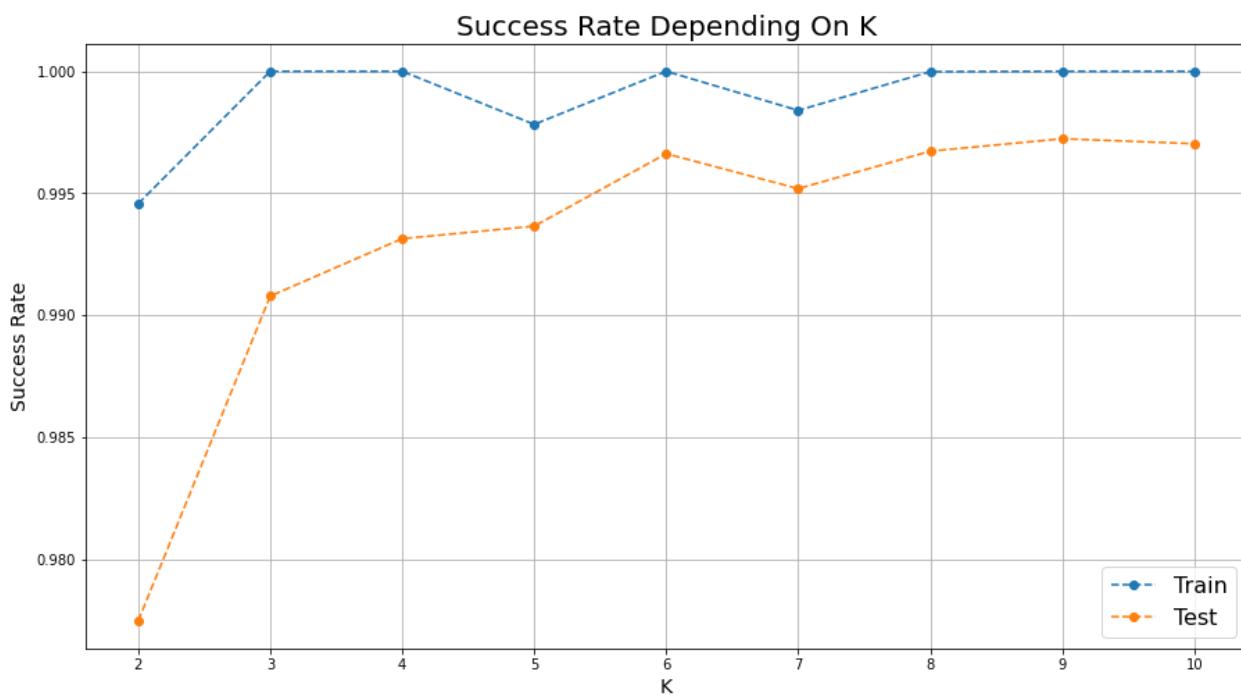
Malware family with the highest false classification (after normalizing):

- Train: Autorun.K
- Test: Autorun.K









Model Summary:

Model: "sequential"

Layer (type)	Output Shape	Param #
<hr/>		
conv2d (Conv2D)	(None, 64, 64, 32)	320
conv2d_1 (Conv2D)	(None, 62, 62, 32)	9248
flatten (Flatten)	(None, 123008)	0
dense (Dense)	(None, 25)	3075225
<hr/>		
Total params: 3,084,793		
Trainable params: 3,084,793		
Non-trainable params: 0		

Train & Test (Includes test results)

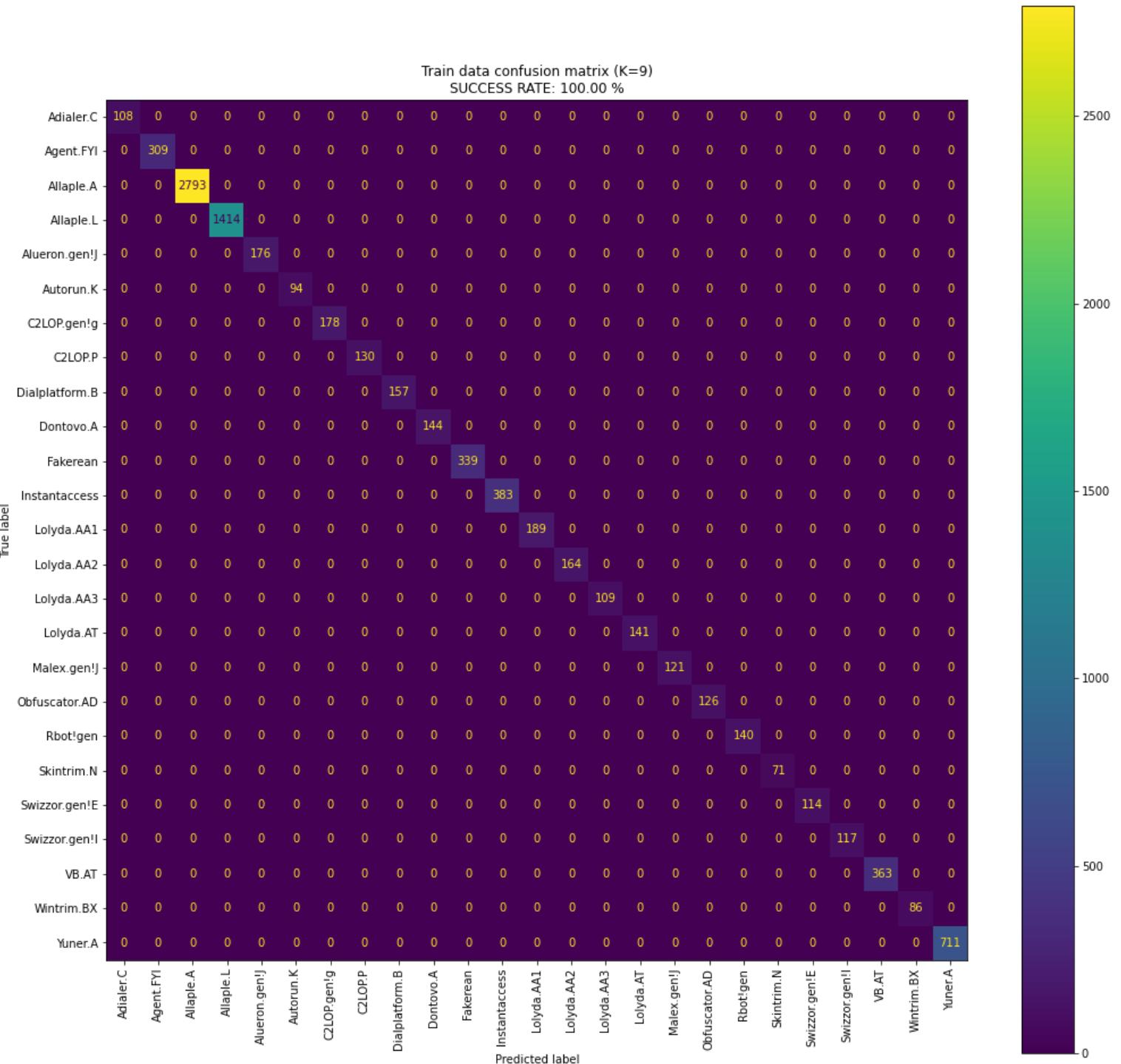
K = 9

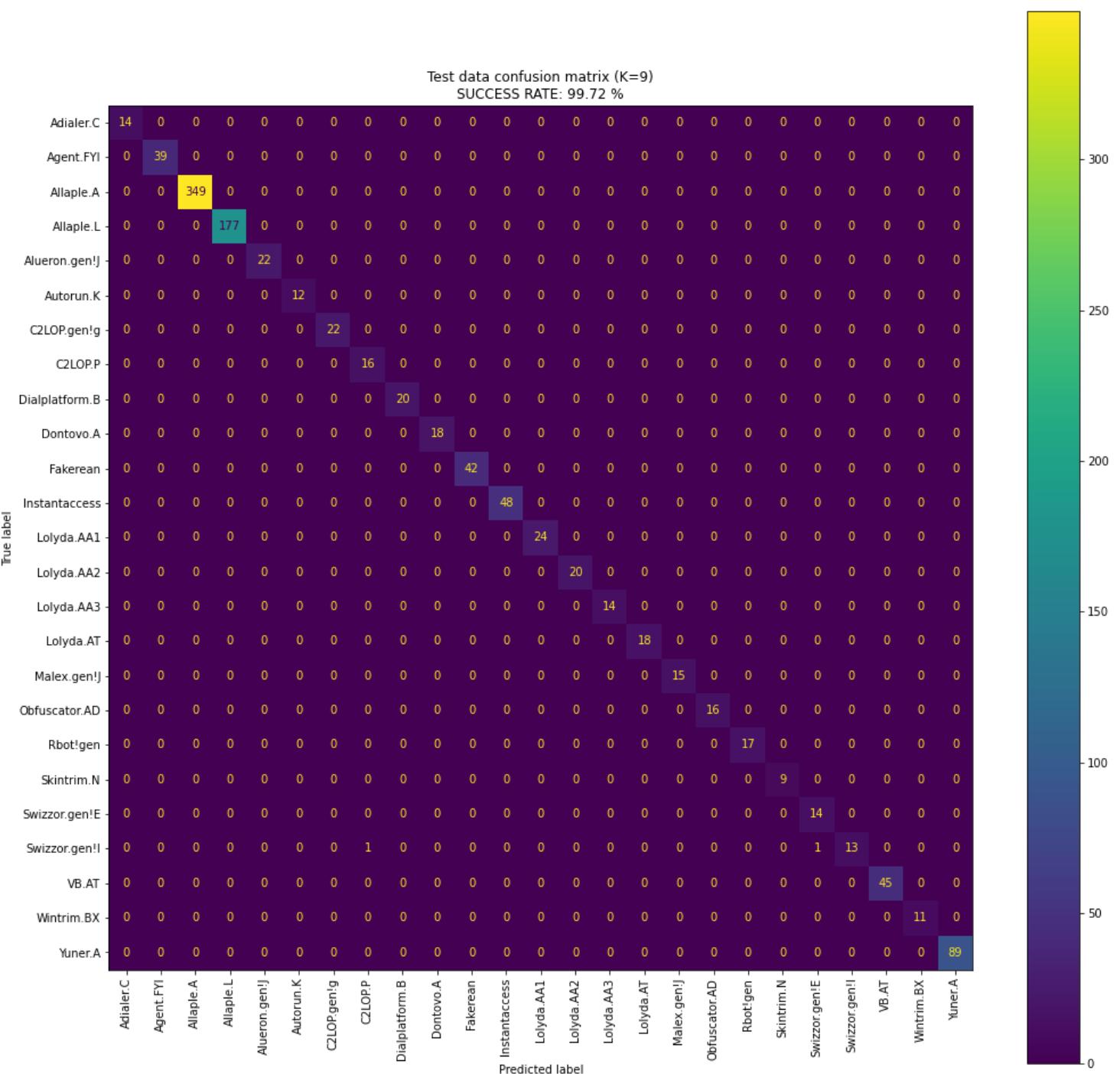
Success Rate (after normalizing):

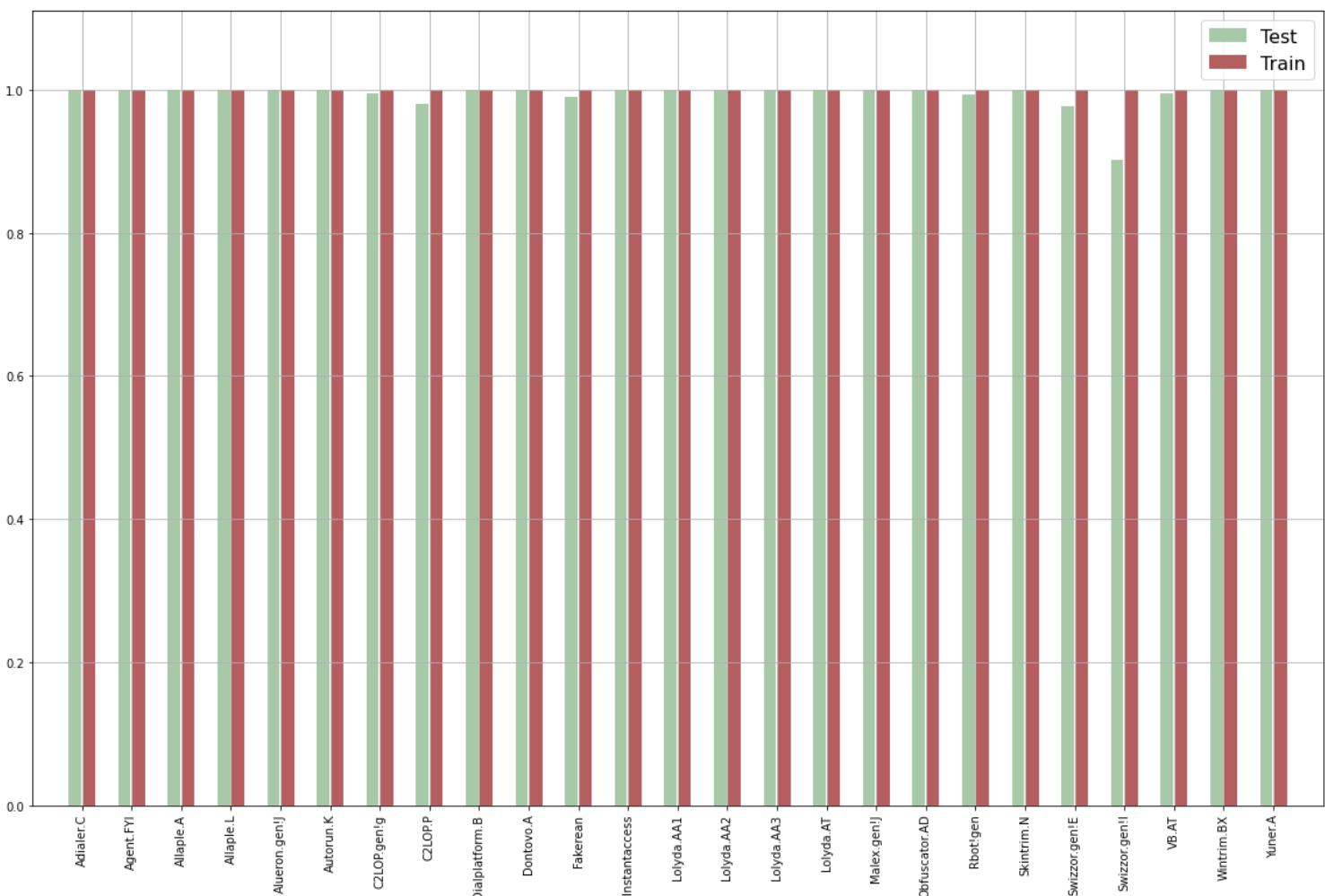
- Train: 100.000 %
- Test: 99.724 %

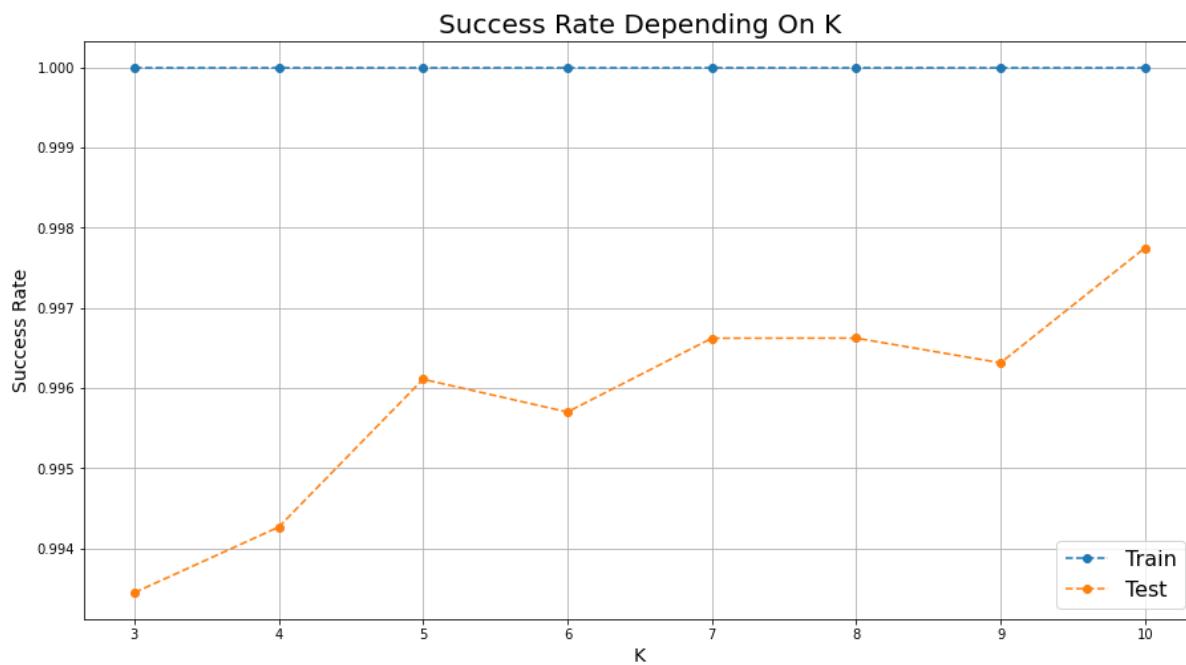
Malware family with the highest false classification (after normalizing):

- Train: N/A
- Test: Swizzor.gen!I









Model Summary:

Model: "sequential"

Layer (type)	Output Shape	Param #
<hr/>		
conv2d (Conv2D)	(None, 128, 128, 32)	320
conv2d_1 (Conv2D)	(None, 126, 126, 32)	9248
flatten (Flatten)	(None, 508032)	0
dense (Dense)	(None, 25)	12700825
<hr/>		
Total params: 12,710,393		
Trainable params: 12,710,393		
Non-trainable params: 0		

Train & Test (Includes test results)

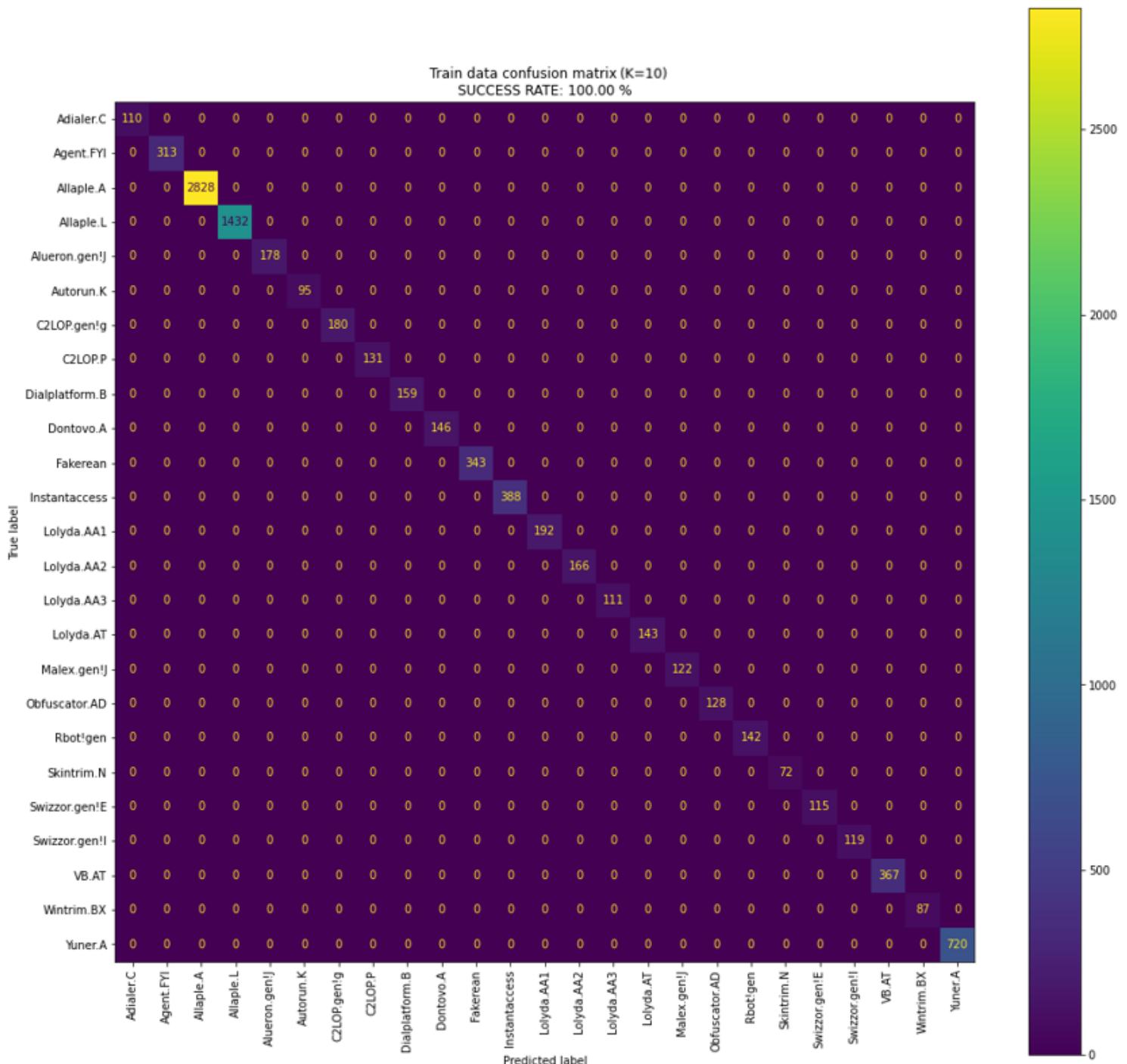
K = 10

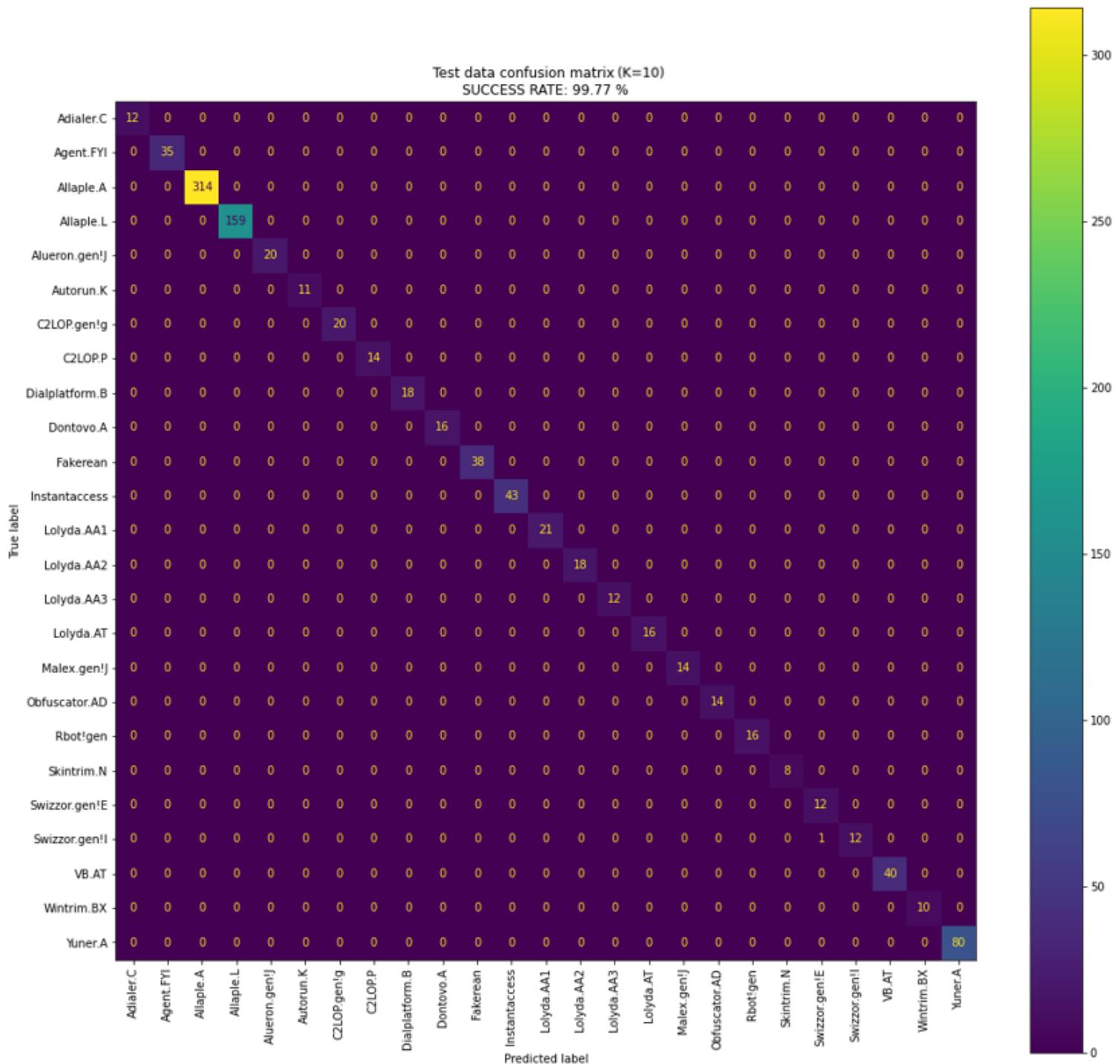
Success Rate (after normalizing):

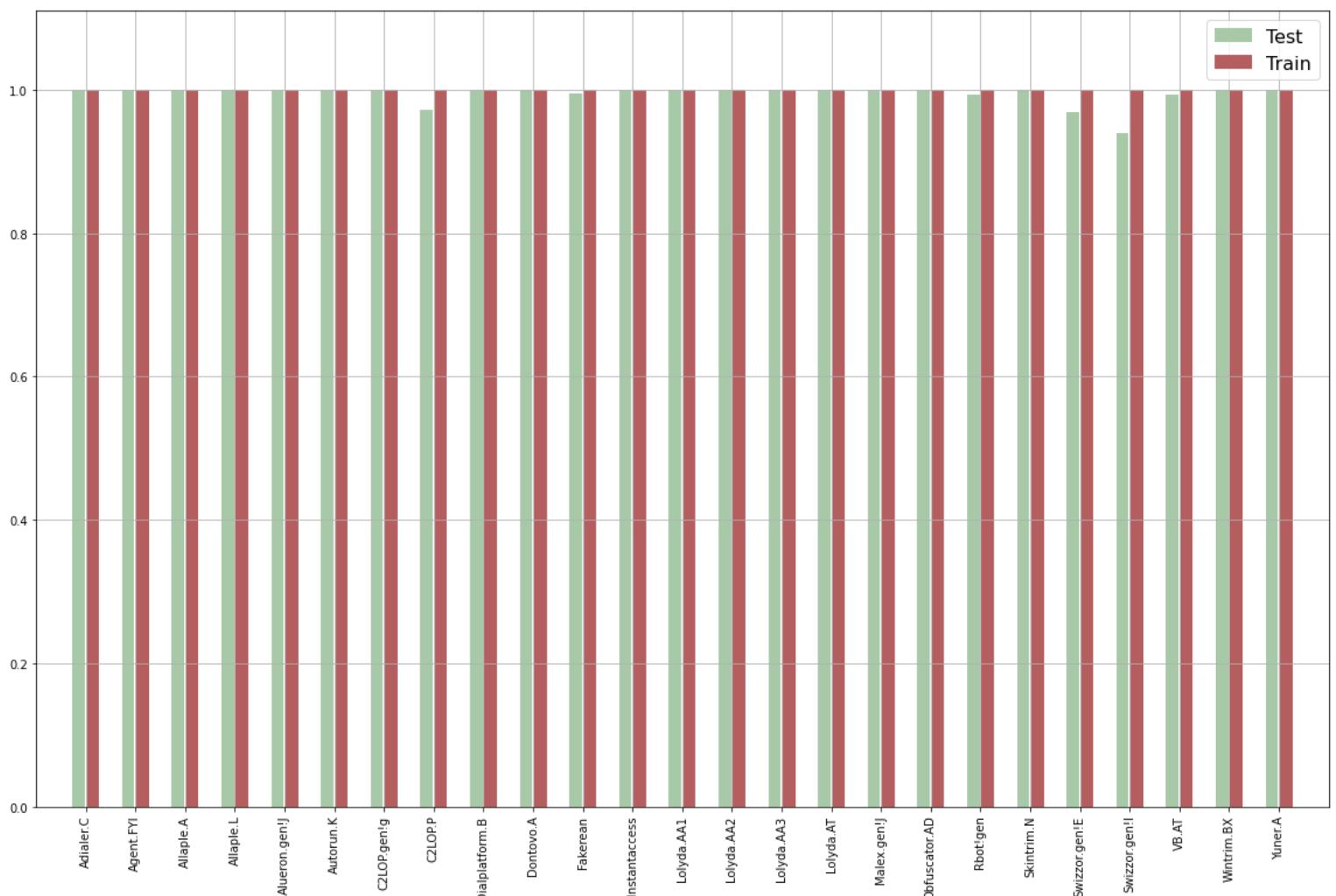
- Train: 100.000 %
- Test: 99.775 %

Malware family with the highest false classification (after normalizing):

- Train: N/A
- Test: Swizzor.gen!I



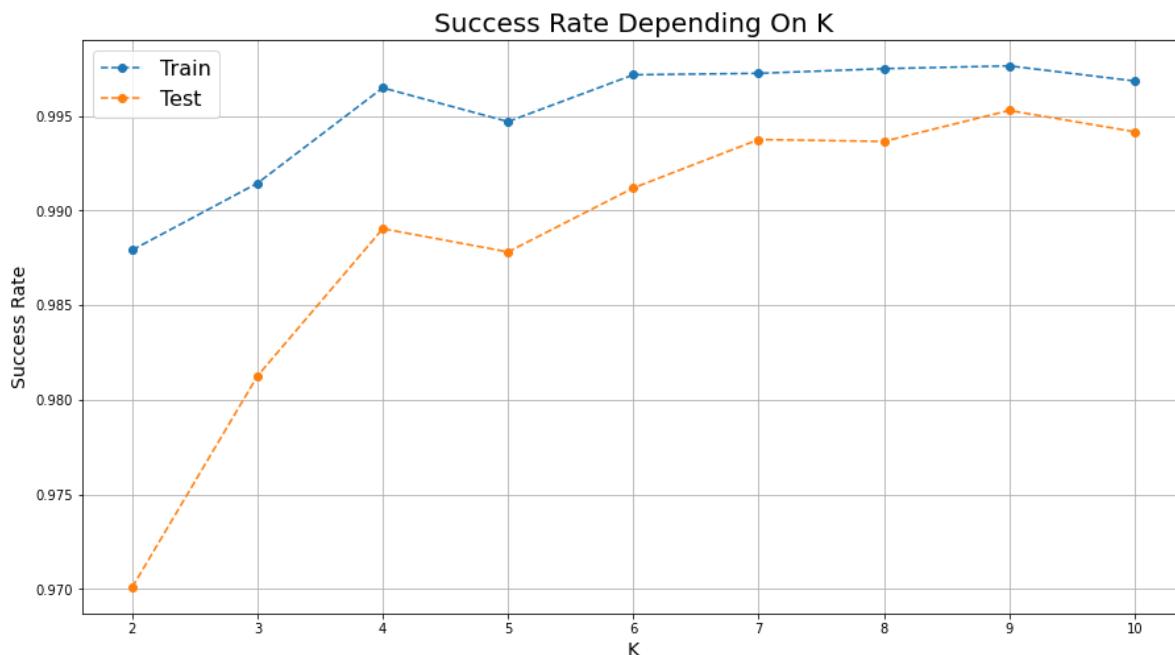




CNN (2 Layers, 64 Filter, 11 Epochs)				
Size	Best K - Test	Test Accuracy	Best K - Train	Train Accuracy
32x32	9	99.529	9	99.764
64x64	10	99.754	10	100
128x128	10	99.734	10	100

טבלה 7: תוצאות CNN שתי שכבות, 64 פילטרים

32×32 5.4.4.1



Model Summary:

Model: "sequential"

```

Layer (type)          Output Shape       Param #
=====
conv2d (Conv2D)      (None, 32, 32, 64)   640
conv2d_1 (Conv2D)     (None, 30, 30, 64)   36928
flatten (Flatten)    (None, 57600)        0
dense (Dense)         (None, 25)           1440025
=====
Total params: 1,477,593
Trainable params: 1,477,593
Non-trainable params: 0
  
```

Train & Test (Includes test results)

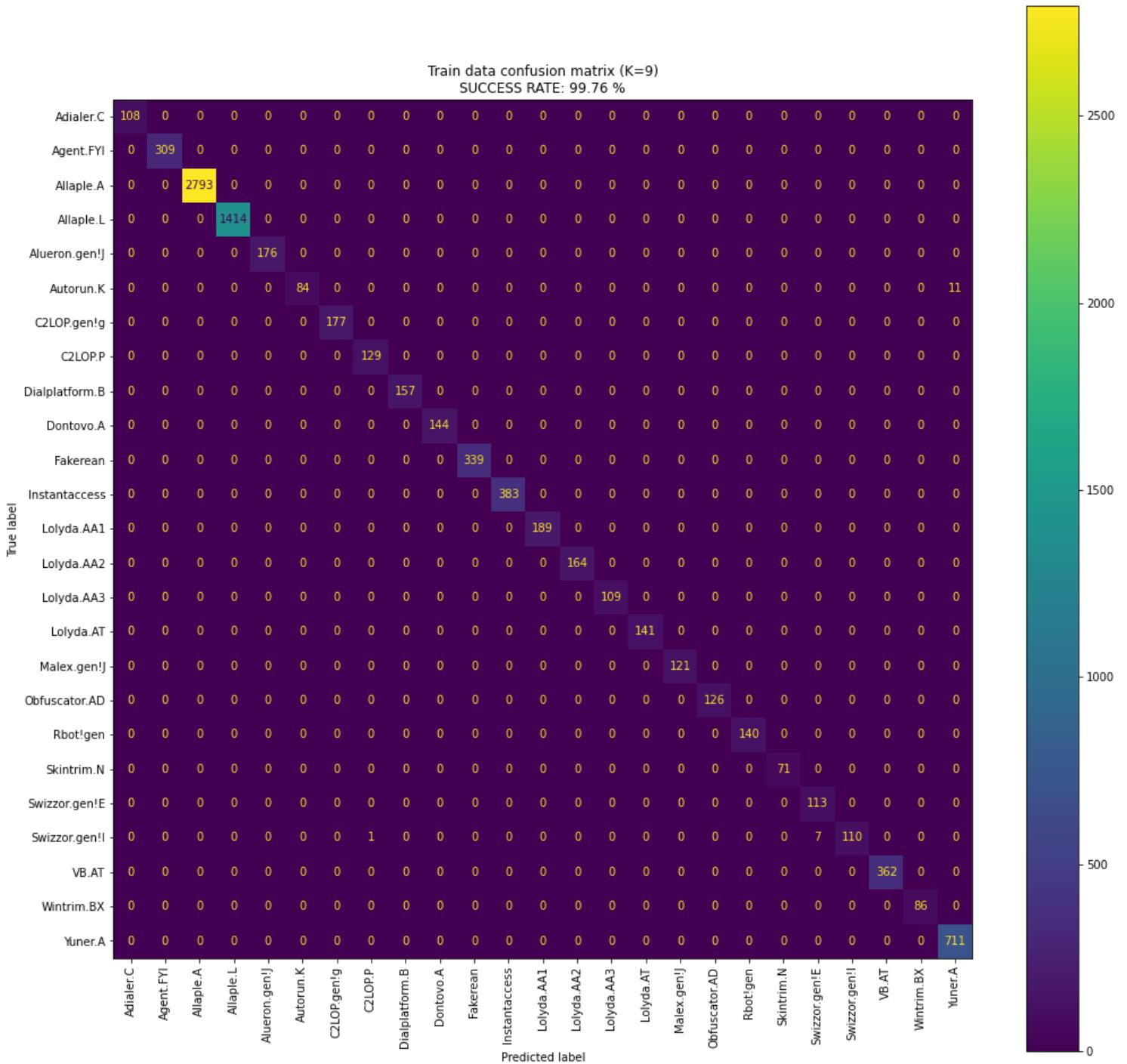
K = 9

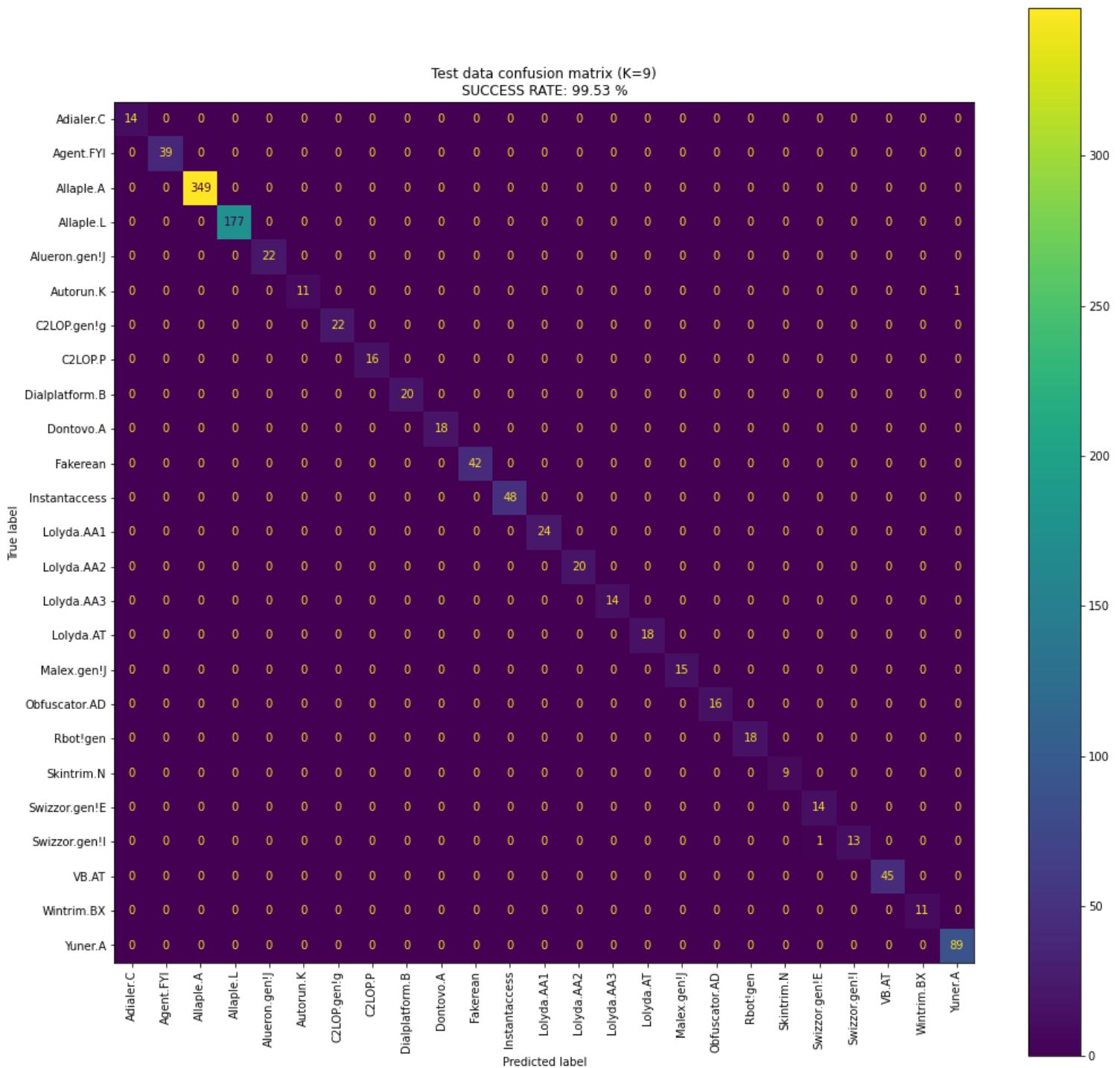
Success Rate (after normalizing):

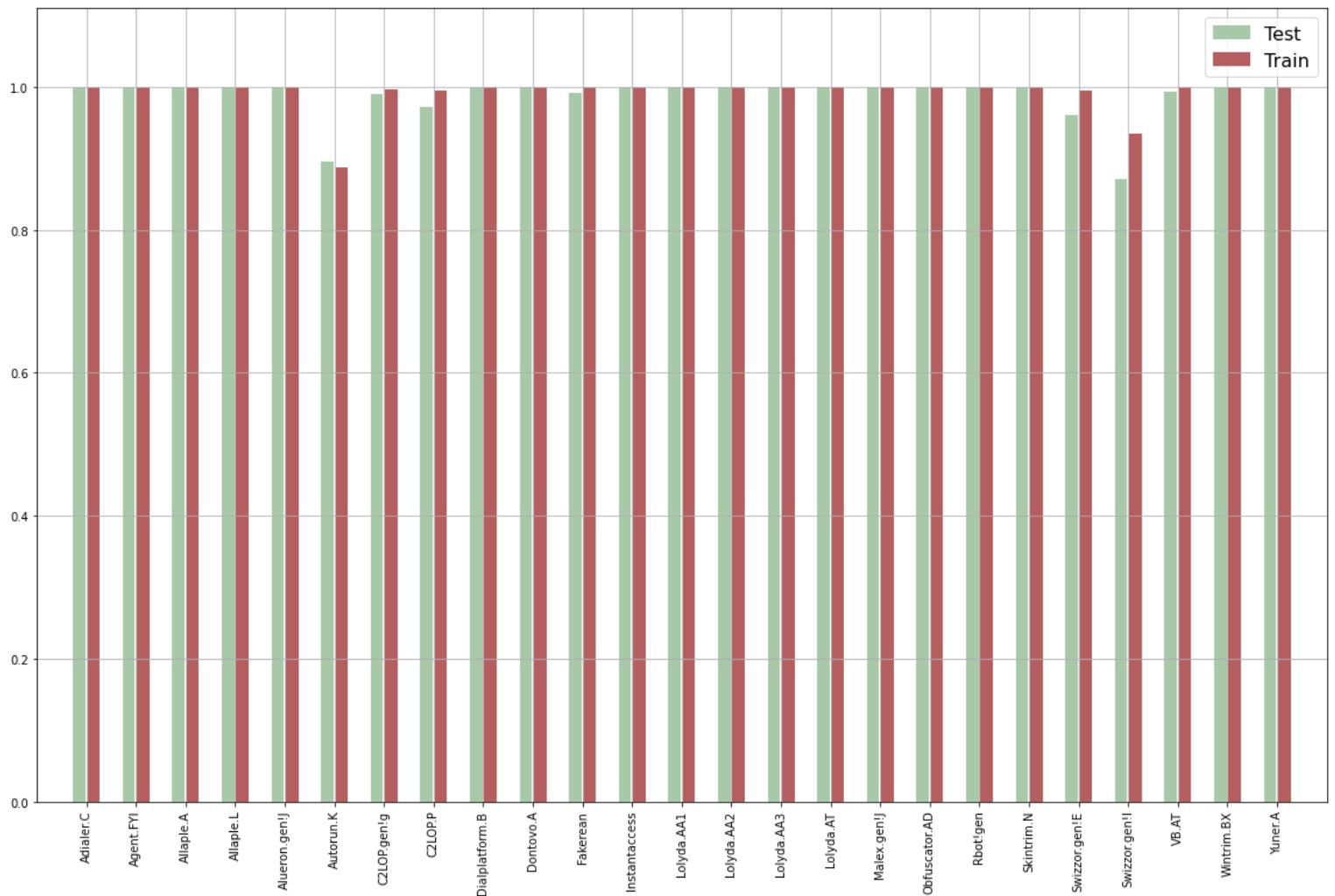
- Train: 99.764 %
- Test: 99.529 %

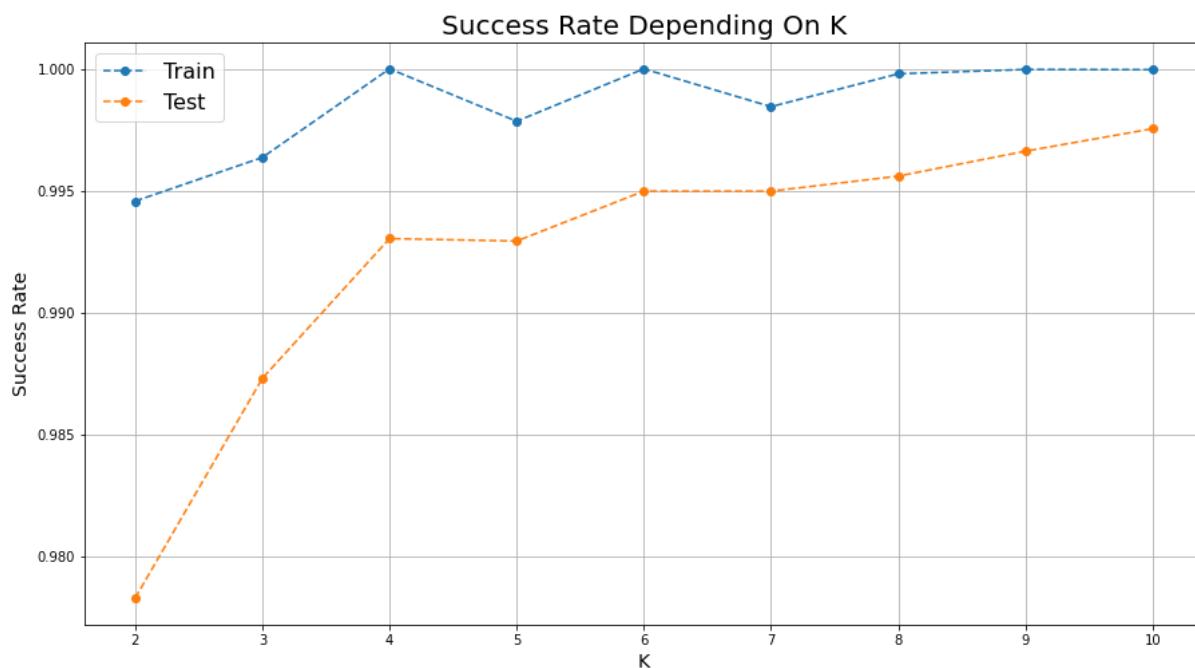
Malware family with the highest false classification (after normalizing):

- Train: Autorun.K
- Test: Swizzor.gen!I









Model Summary:

Model: "sequential"

Layer (type)	Output Shape	Param #
<hr/>		
conv2d (Conv2D)	(None, 64, 64, 64)	640
conv2d_1 (Conv2D)	(None, 62, 62, 64)	36928
flatten (Flatten)	(None, 246016)	0
dense (Dense)	(None, 25)	6150425
<hr/>		
Total params: 6,187,993		
Trainable params: 6,187,993		
Non-trainable params: 0		

Train & Test (Includes test results)

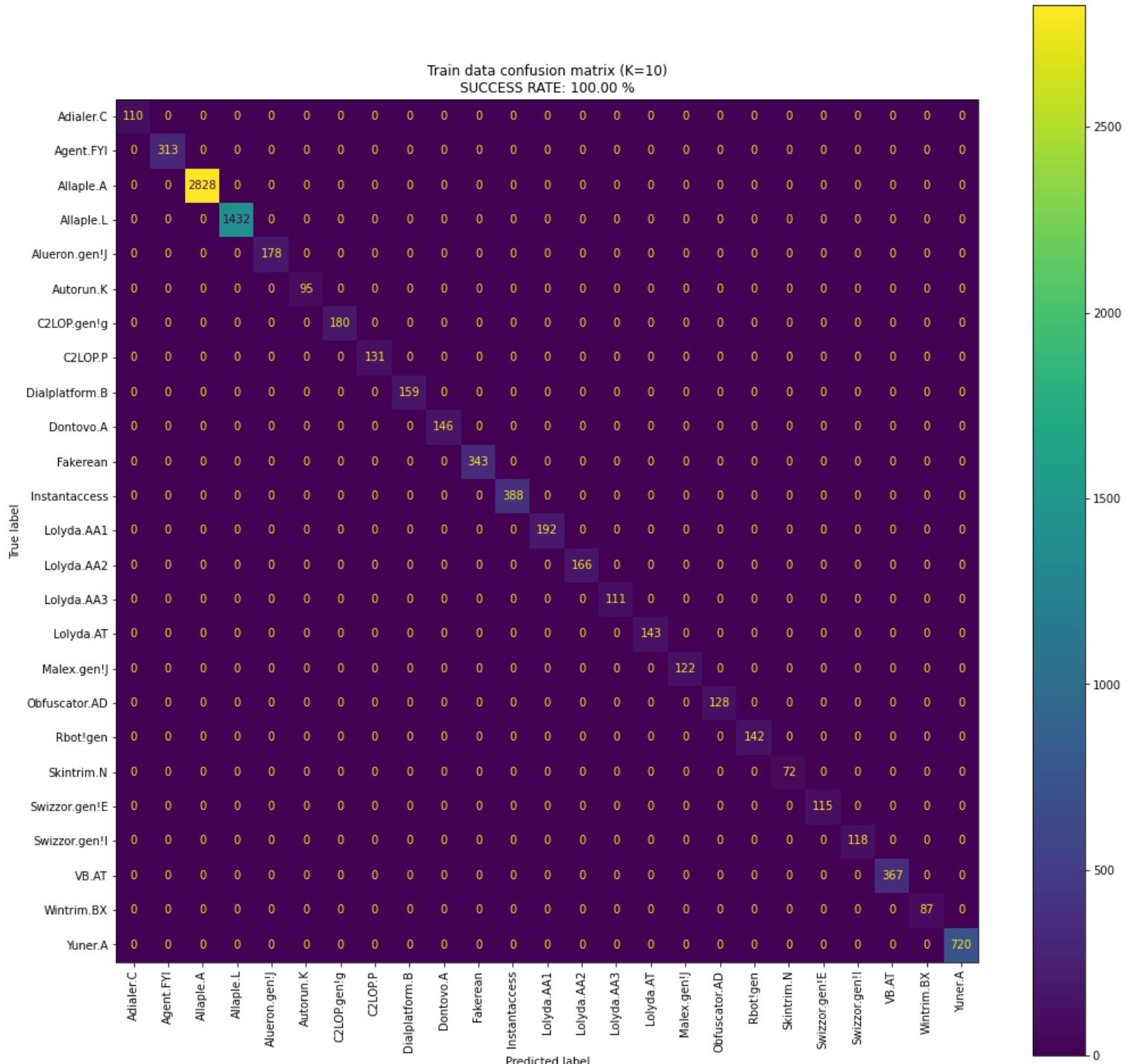
K = 10

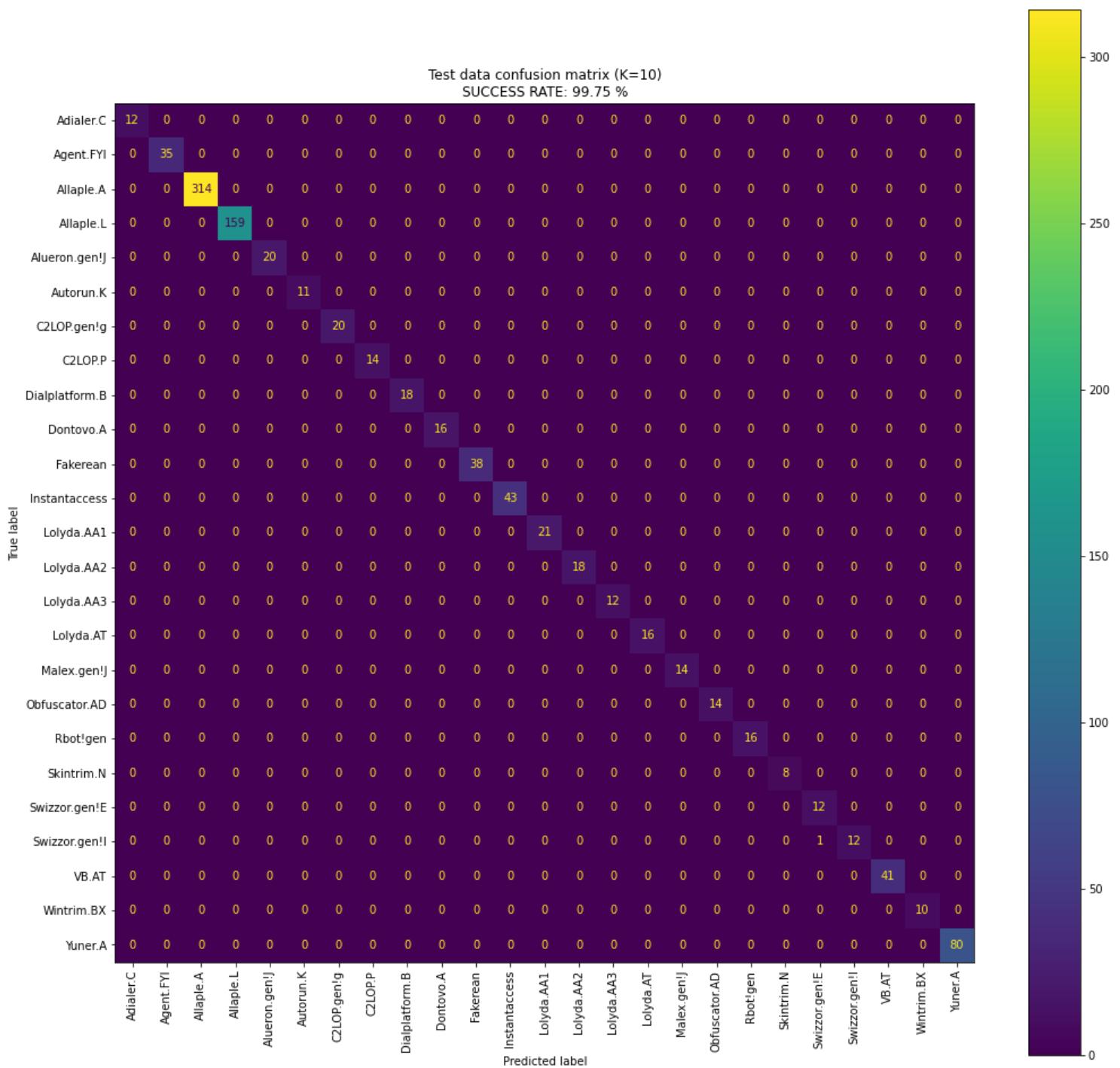
Success Rate (after normalizing):

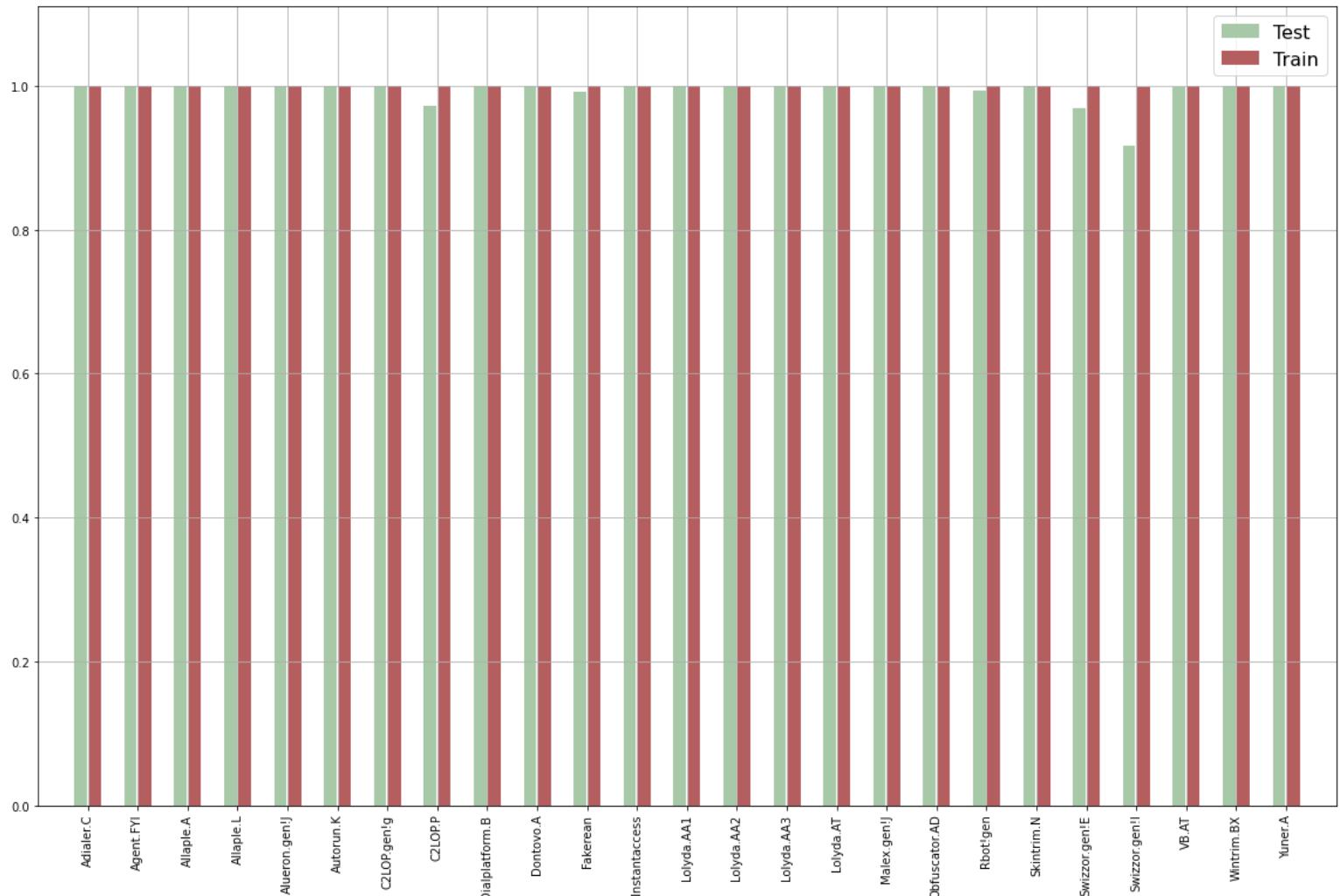
- Train: 99.997 %
- Test: 99.754 %

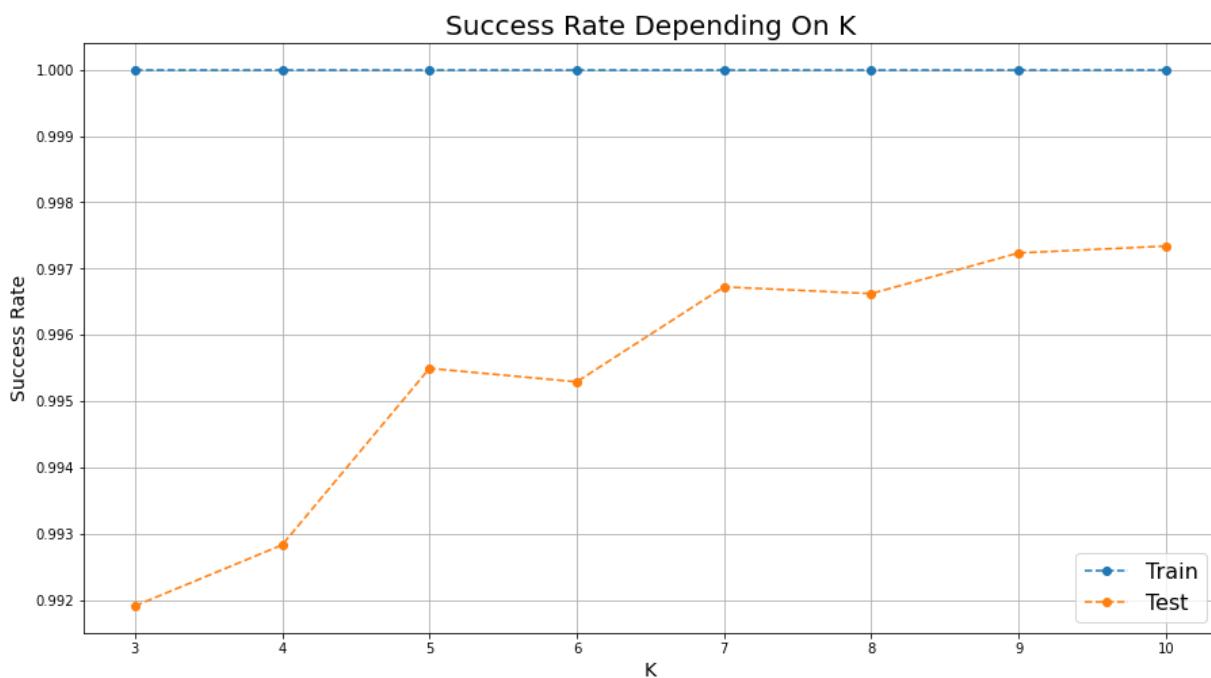
Malware family with the highest false classification (after normalizing):

- Train: Swizzor.gen!I
- Test: Swizzor.gen!I









Model Summary:

Model: "sequential"

Layer (type)	Output Shape	Param #
<hr/>		
conv2d (Conv2D)	(None, 128, 128, 64)	640
conv2d_1 (Conv2D)	(None, 126, 126, 64)	36928
flatten (Flatten)	(None, 1016064)	0
dense (Dense)	(None, 25)	25401625
<hr/>		
Total params: 25,439,193		
Trainable params: 25,439,193		
Non-trainable params: 0		

Train & Test (Includes test results)

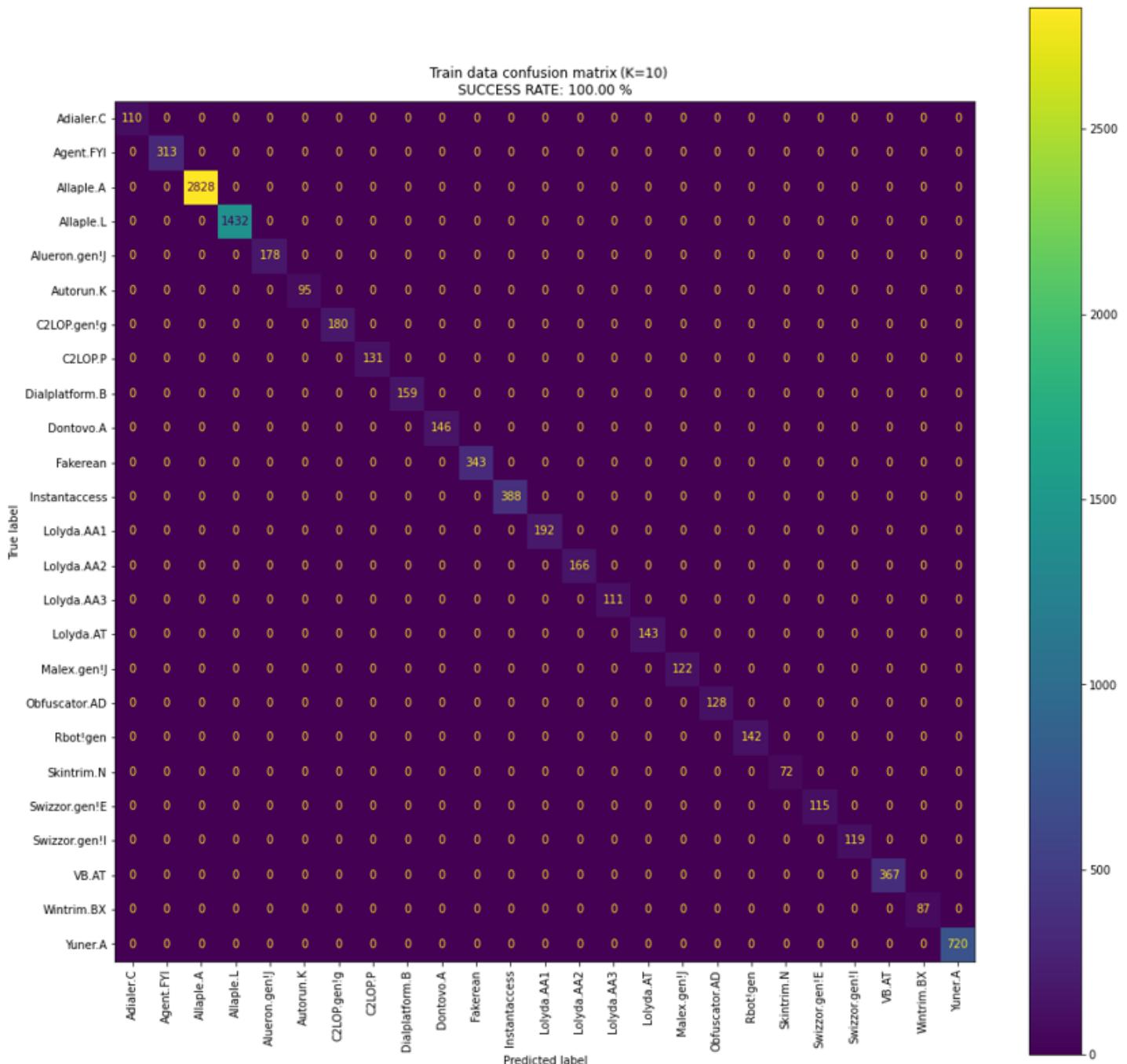
K = 10

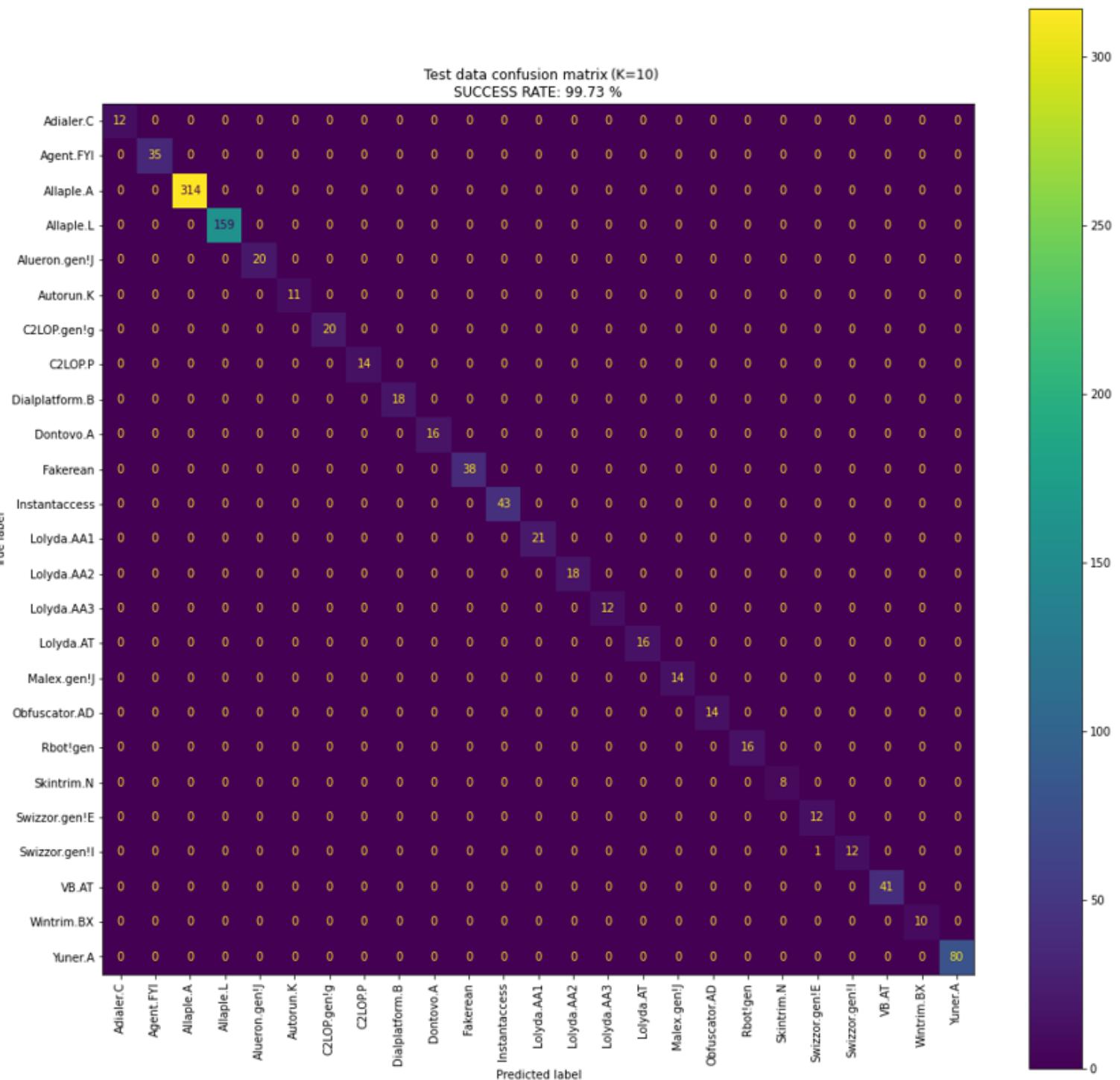
Success Rate (after normalizing):

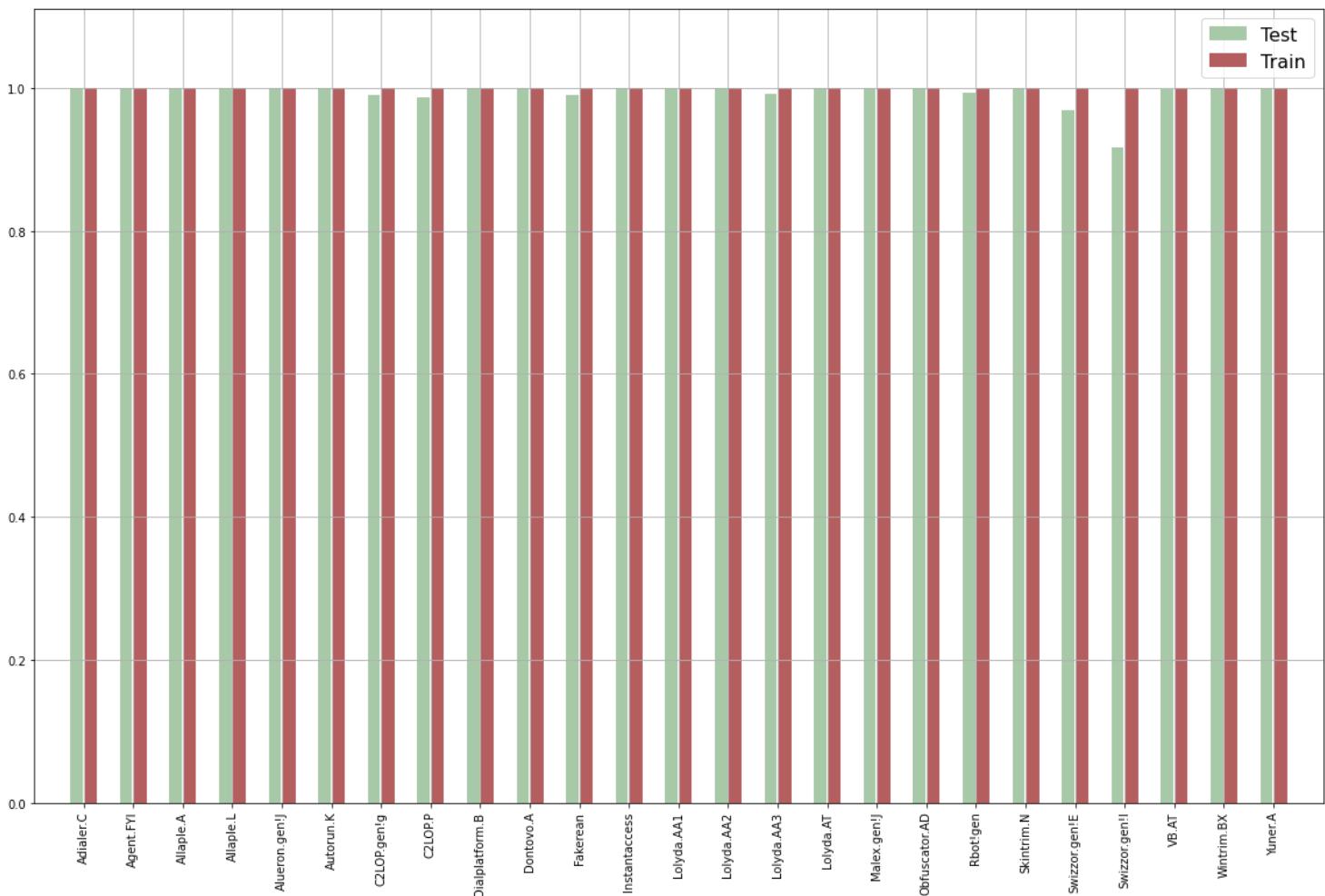
- Train: 100.000 %
- Test: 99.734 %

Malware family with the highest false classification (after normalizing):

- Train: N/A
- Test: Swizzor.gen!I





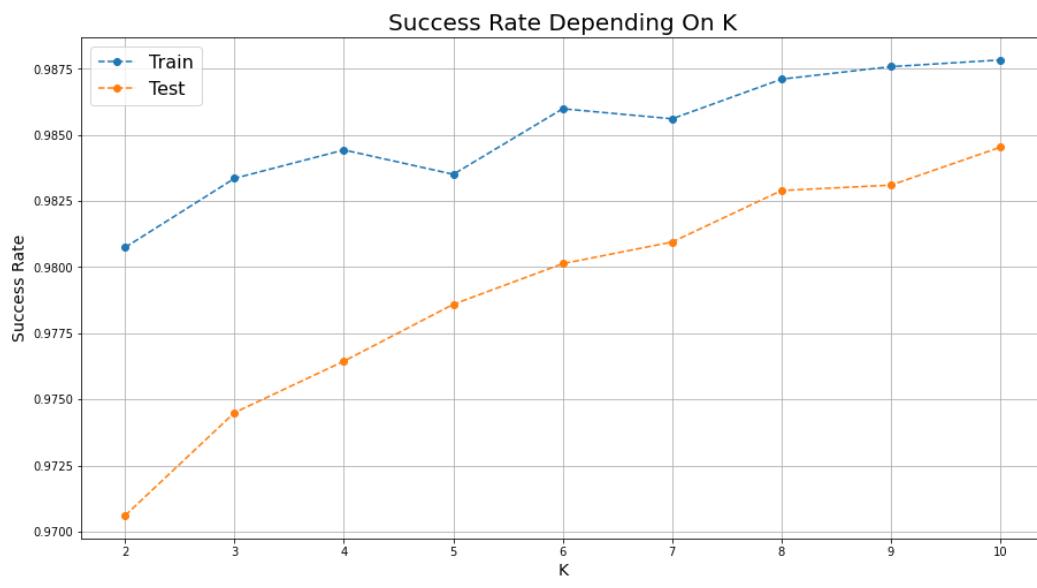


מודל CNN מורכב 5.4.5

CNN (Complex Model, 11 Epochs, 100 Batch Size, <size> Filter)				
Size	Best K - Test	Test Accuracy	Best K - Train	Train Accuracy
32x32	10	98.525	10	98.804
64x64	10	98.833	10	99.107
128x128	9	99.109	7	99.447

טבלה 8: תוצאות CNN מודל מורכב

32×32 5.4.5.1



Model Summary:

Model: "sequential"

Layer (type)	Output Shape	Param #
conv2d (Conv2D)	(None, 32, 32, 32)	320
conv2d_1 (Conv2D)	(None, 30, 30, 32)	9248
max_pooling2d (MaxPooling2D)	(None, 15, 15, 32)	0
)		
dropout (Dropout)	(None, 15, 15, 32)	0
conv2d_2 (Conv2D)	(None, 15, 15, 64)	18496
conv2d_3 (Conv2D)	(None, 13, 13, 64)	36928
max_pooling2d_1 (MaxPooling2D)	(None, 6, 6, 64)	0
)		
dropout_1 (Dropout)	(None, 6, 6, 64)	0
flatten (Flatten)	(None, 2304)	0
dense (Dense)	(None, 512)	1180160
dense_1 (Dense)	(None, 512)	262656
dropout_2 (Dropout)	(None, 512)	0
dense_2 (Dense)	(None, 25)	12825

Total params: 1,520,633
 Trainable params: 1,520,633
 Non-trainable params: 0

Train & Test (Includes test results)

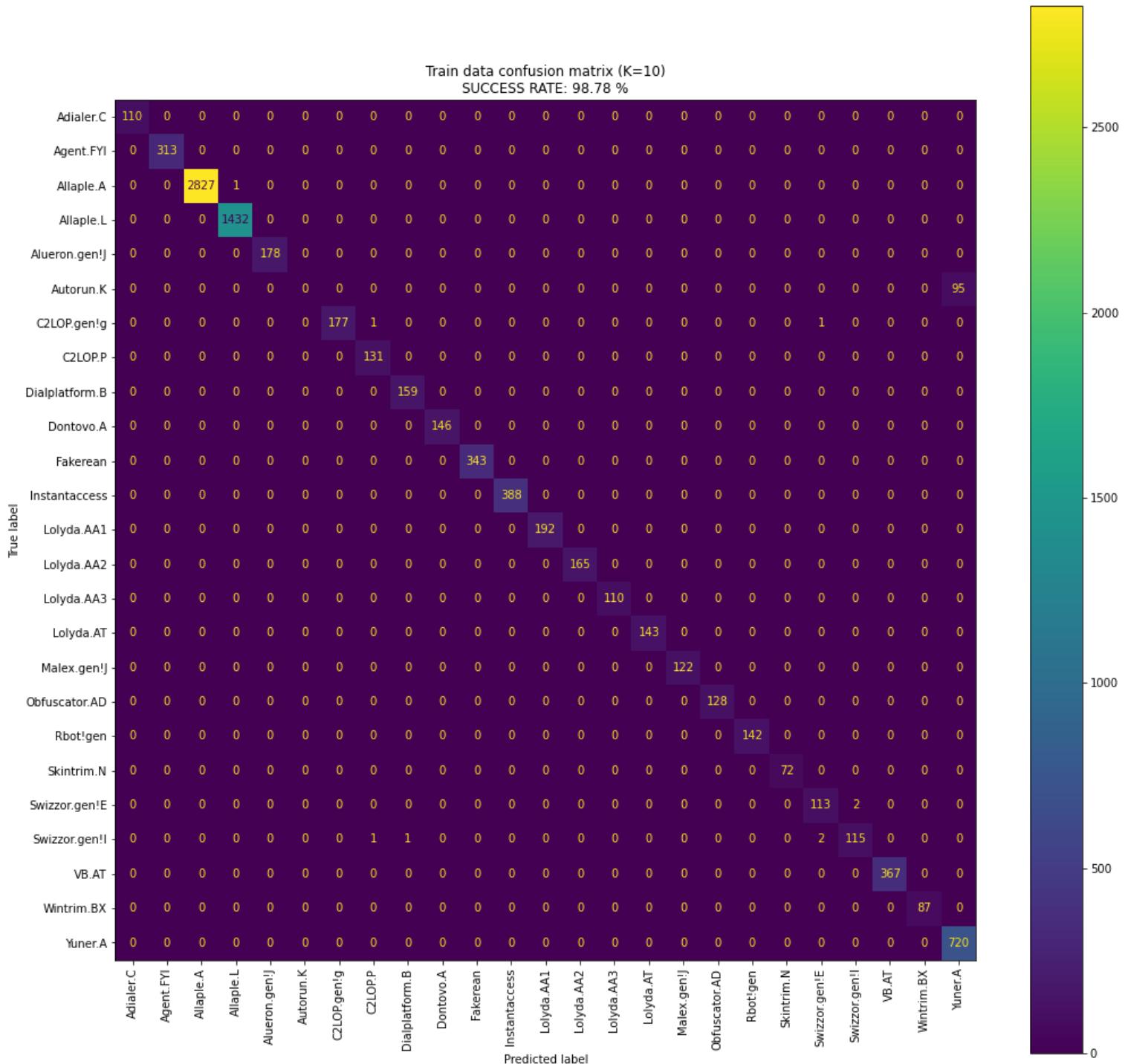
K = 10

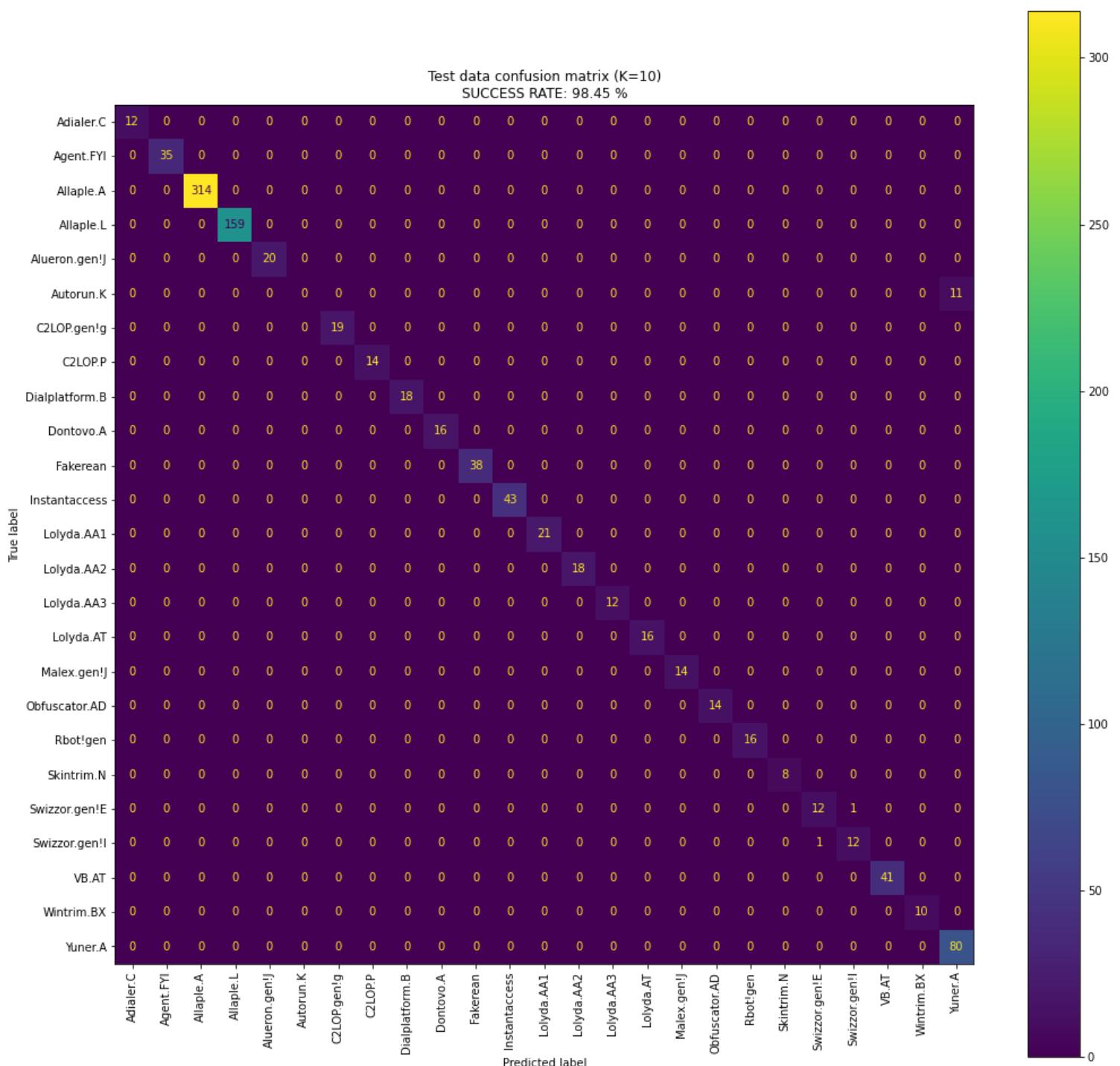
Success Rate (after normalizing):

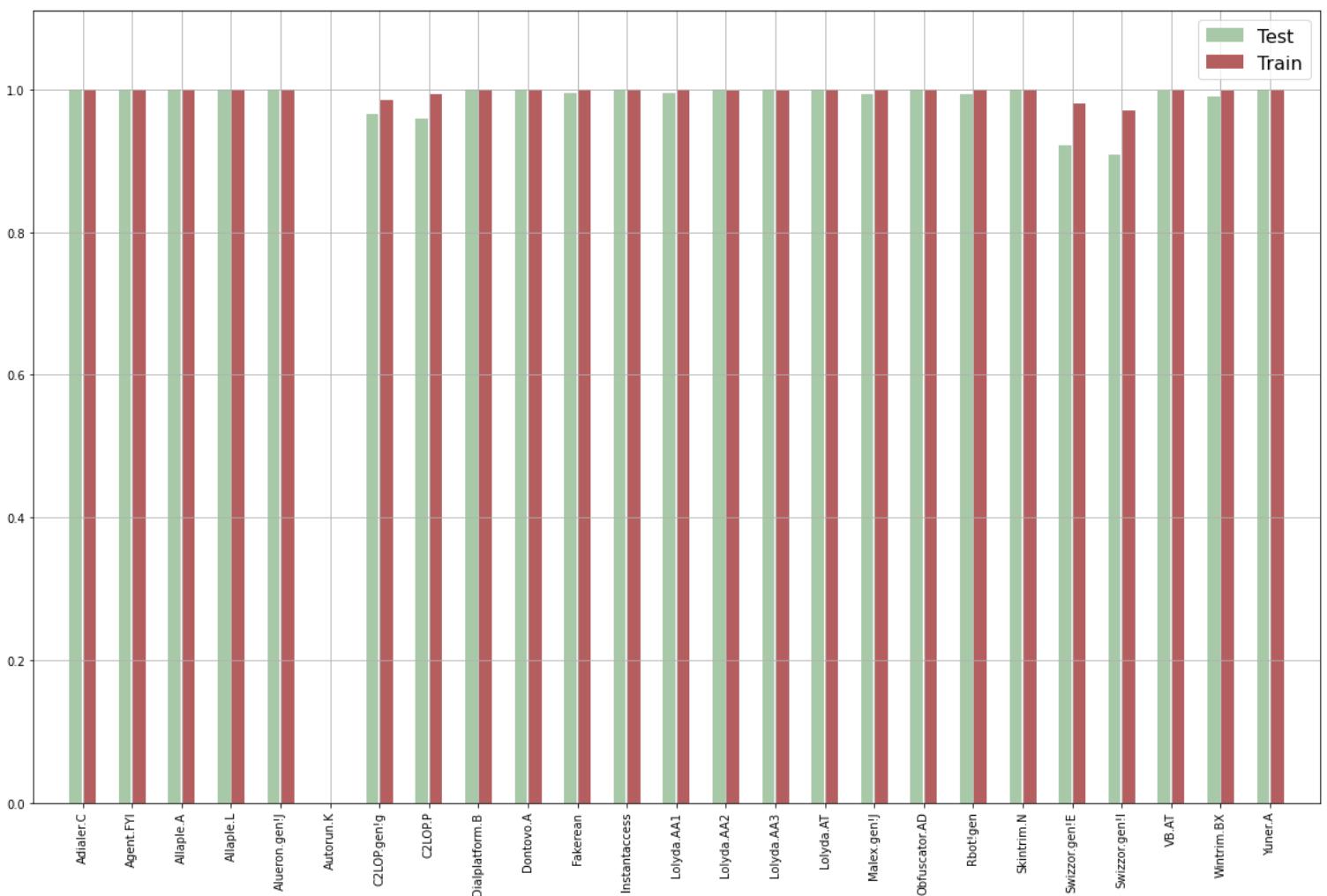
- Train: 98.784 %
- Test: 98.454 %

Malware family with the highest false classification (after normalizing):

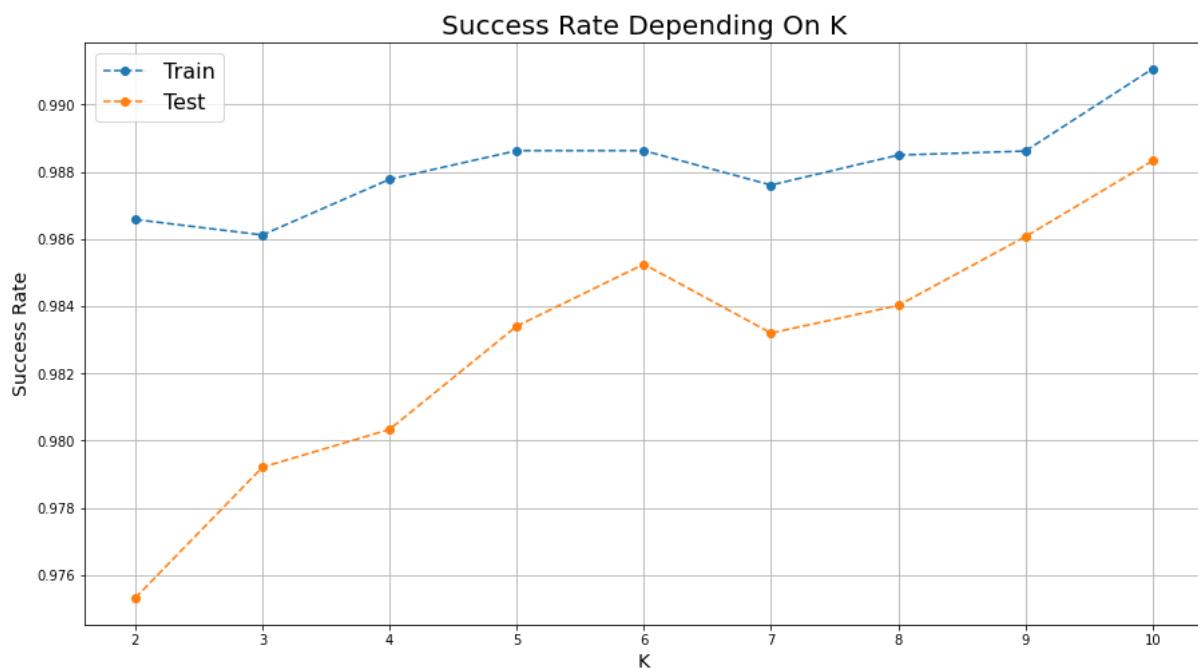
- Train: Autorun.K
- Test: Autorun.K







64 × 64 5.4.5.2



Model Summary:

Model: "sequential"

Layer (type)	Output Shape	Param #
<hr/>		
conv2d (Conv2D)	(None, 64, 64, 64)	640
conv2d_1 (Conv2D)	(None, 62, 62, 32)	18464
max_pooling2d (MaxPooling2D)	(None, 31, 31, 32)	0
dropout (Dropout)	(None, 31, 31, 32)	0
conv2d_2 (Conv2D)	(None, 31, 31, 64)	18496
conv2d_3 (Conv2D)	(None, 29, 29, 64)	36928
max_pooling2d_1 (MaxPooling 2D)	(None, 14, 14, 64)	0
dropout_1 (Dropout)	(None, 14, 14, 64)	0
flatten (Flatten)	(None, 12544)	0
dense (Dense)	(None, 512)	6423040
dropout_2 (Dropout)	(None, 512)	0
dense_1 (Dense)	(None, 25)	12825
<hr/>		
Total params:	6,510,393	
Trainable params:	6,510,393	
Non-trainable params:	0	

Train & Test (Includes test results)

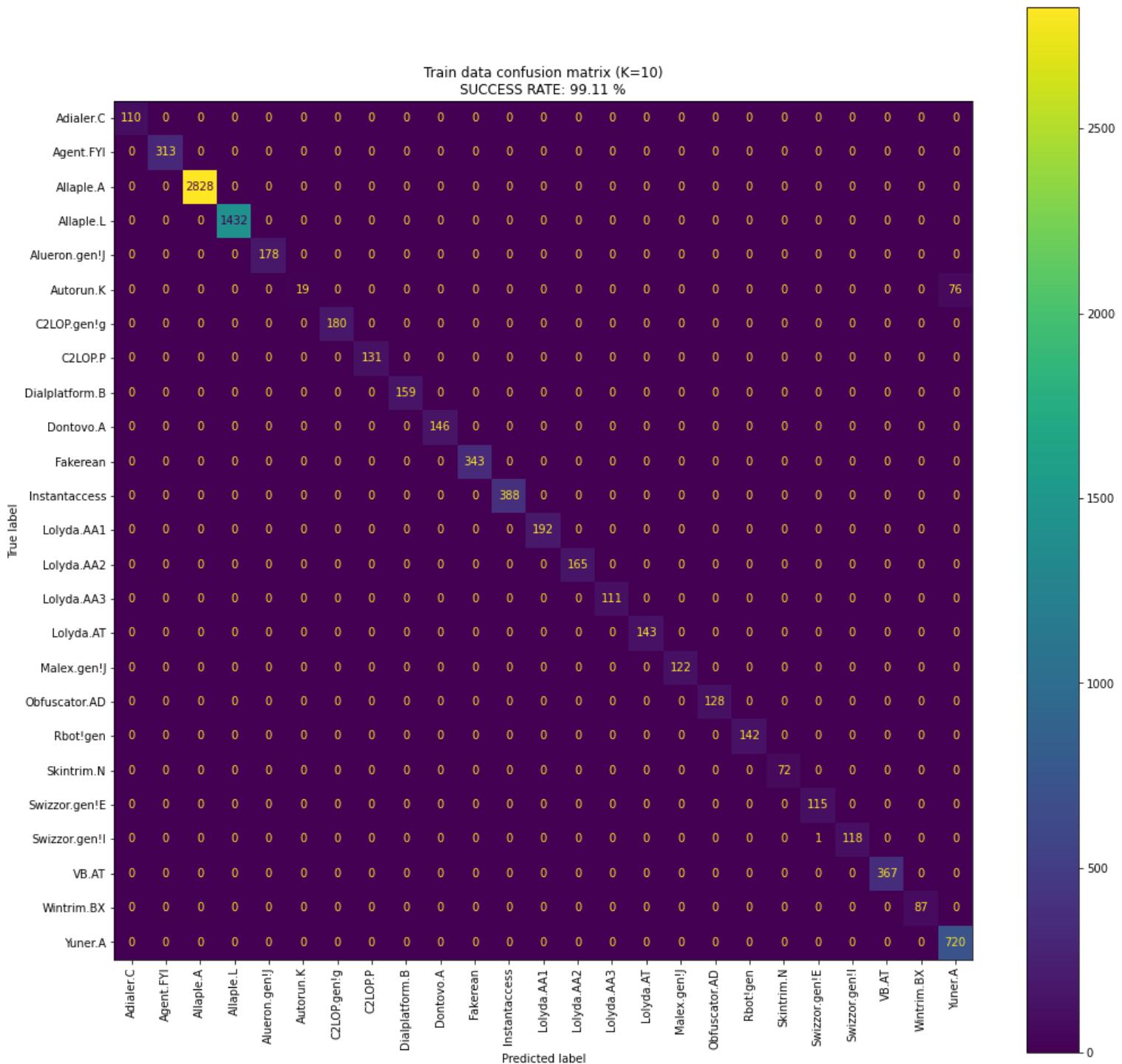
K = 10

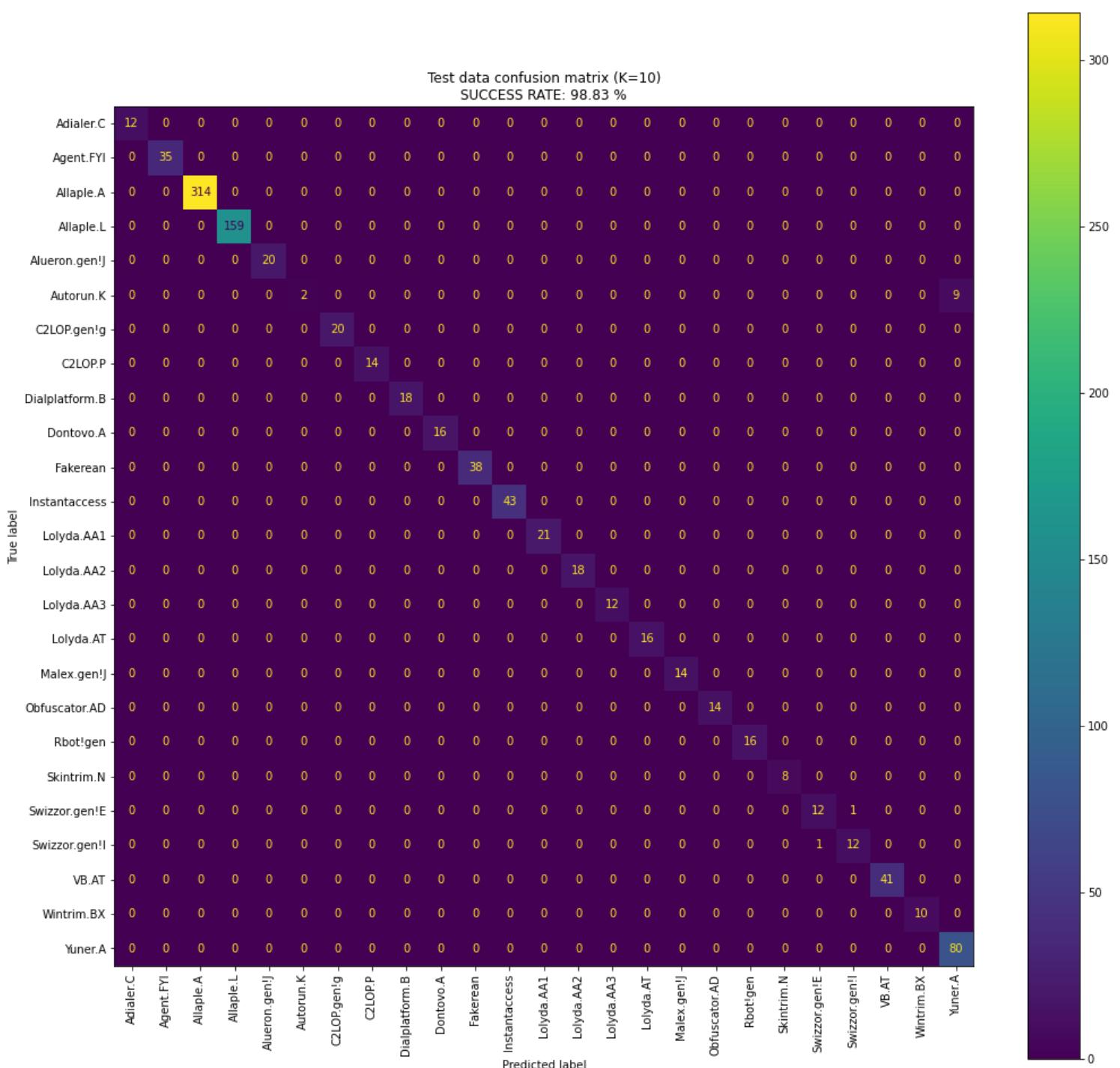
Success Rate (after normalizing):

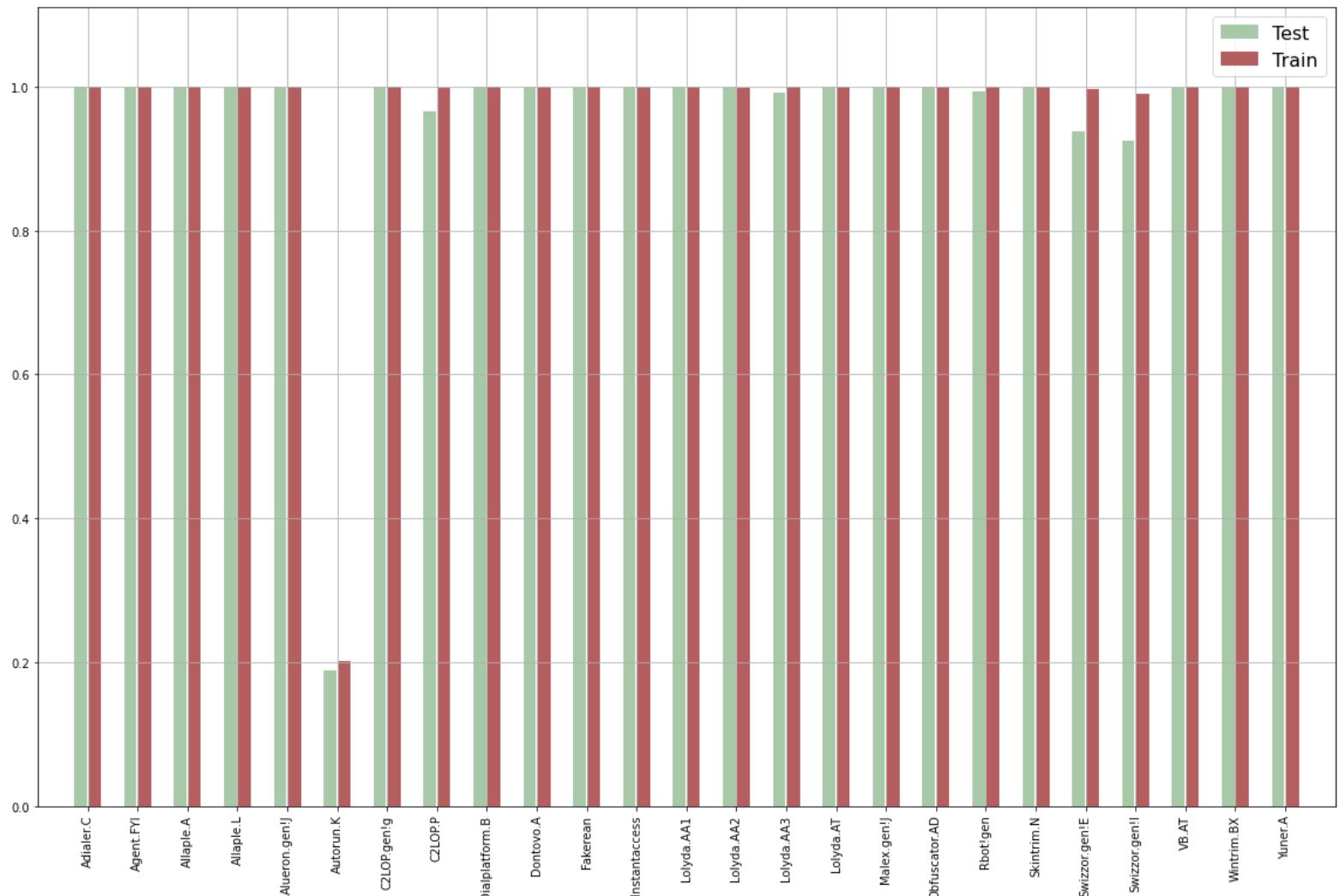
- Train: 99.107 %
- Test: 98.833 %

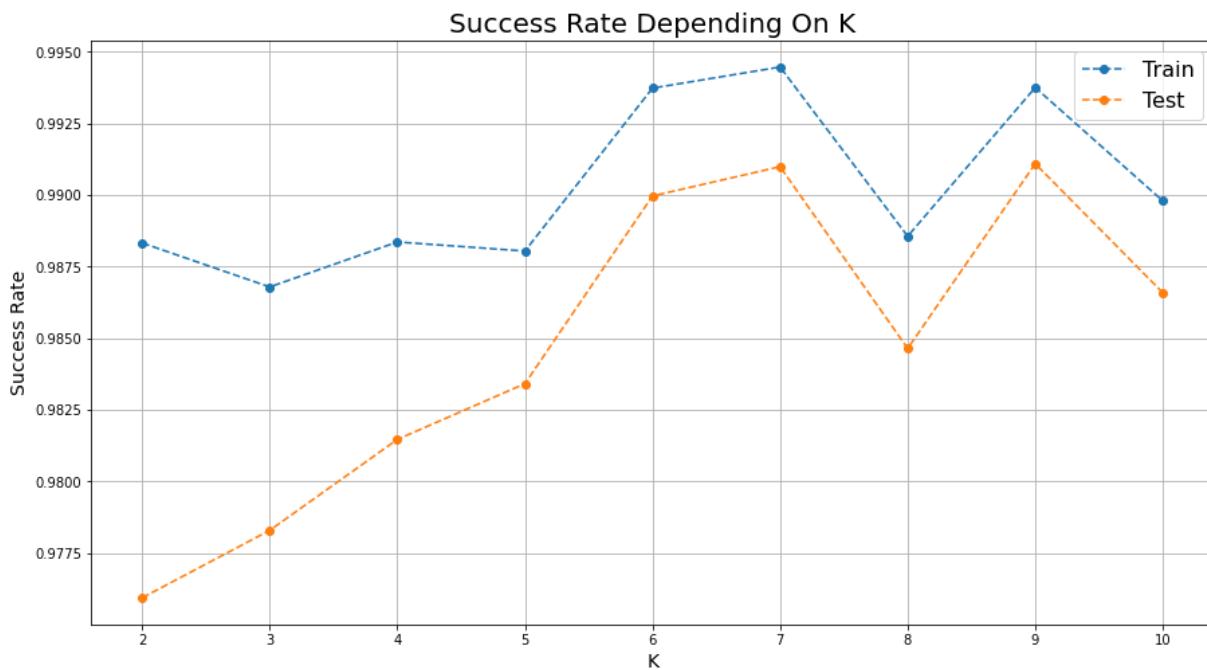
Malware family with the highest false classification (after normalizing):

- Train: Autorun.K
- Test: Autorun.K









Model Summary:

Model: "sequential"

Layer (type)	Output Shape	Param #
<hr/>		
conv2d (Conv2D)	(None, 128, 128, 128)	1280
conv2d_1 (Conv2D)	(None, 126, 126, 32)	36896
max_pooling2d (MaxPooling2D)	(None, 63, 63, 32)	0
dropout (Dropout)	(None, 63, 63, 32)	0
conv2d_2 (Conv2D)	(None, 63, 63, 64)	18496
conv2d_3 (Conv2D)	(None, 61, 61, 64)	36928
max_pooling2d_1 (MaxPooling 2D)	(None, 30, 30, 64)	0
dropout_1 (Dropout)	(None, 30, 30, 64)	0
flatten (Flatten)	(None, 57600)	0
dense (Dense)	(None, 512)	29491712
dropout_2 (Dropout)	(None, 512)	0
dense_1 (Dense)	(None, 25)	12825
<hr/>		
Total params:	29,598,137	
Trainable params:	29,598,137	
Non-trainable params:	0	

Train (Includes test results)

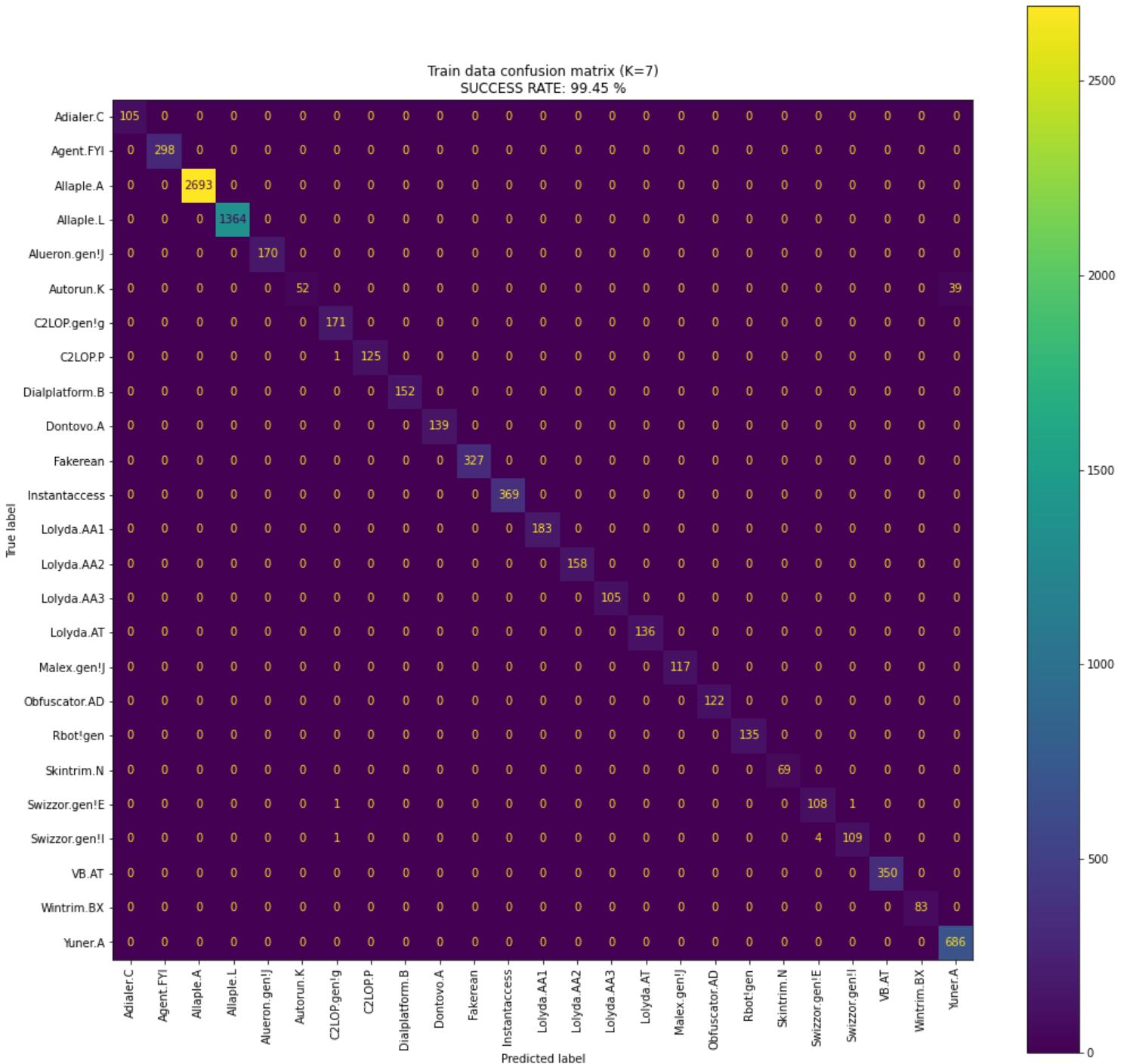
K = 7

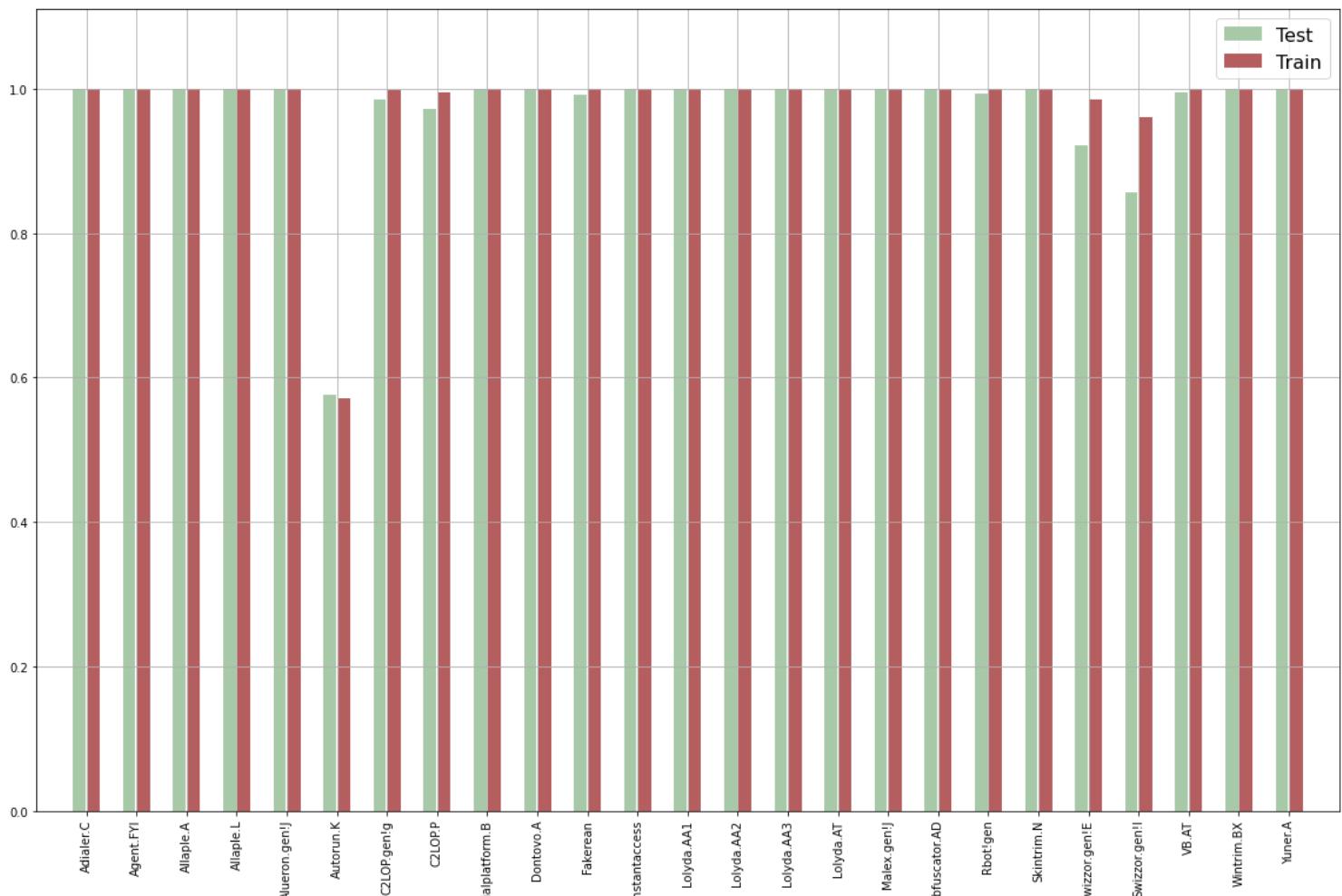
Success Rate (after normalizing):

- Train: 99.447 %
- Test: 99.099 %

Malware family with the highest false classification (after normalizing):

- Train: Autorun.K
 - Test: Autorun.K





Test (Includes train results)

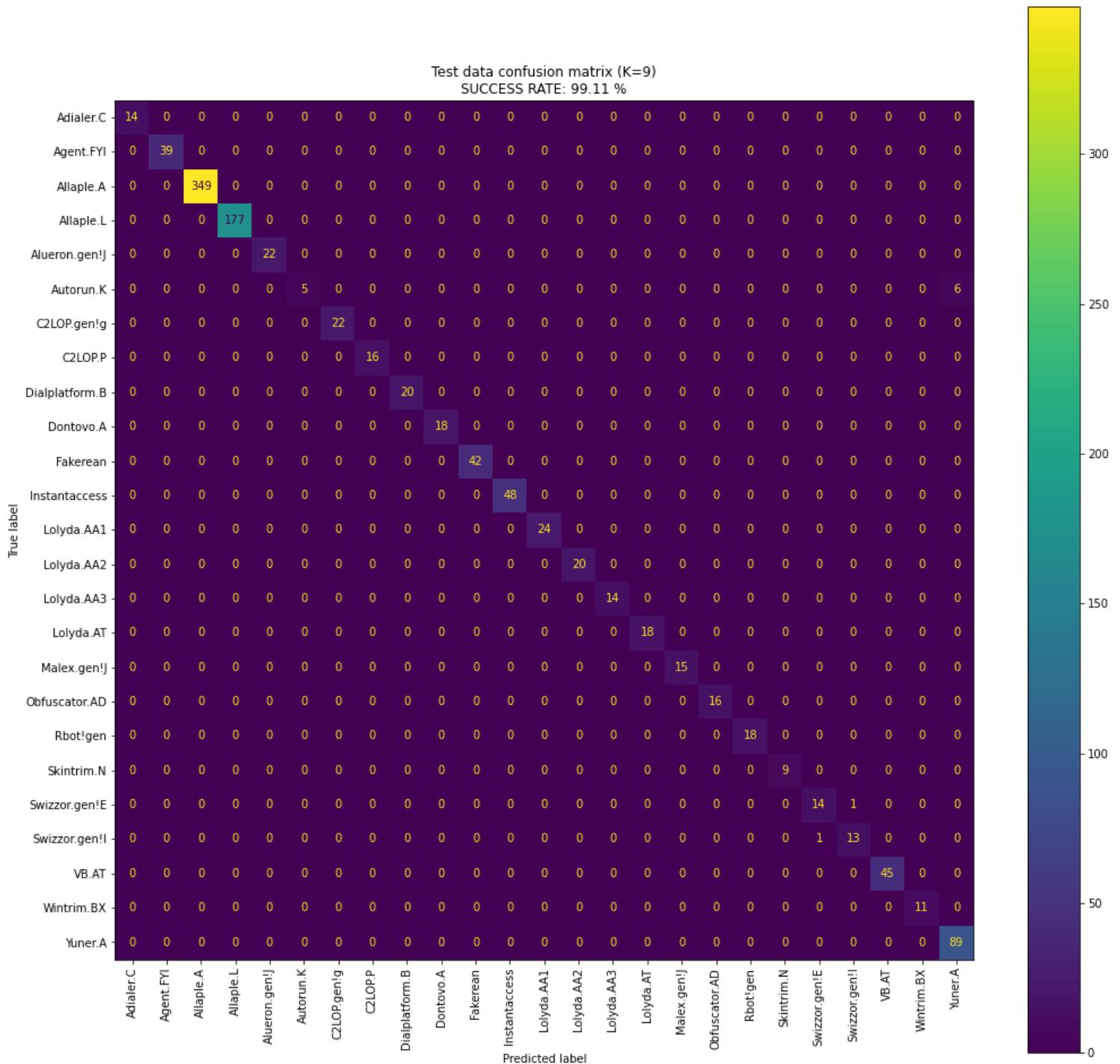
K = 9

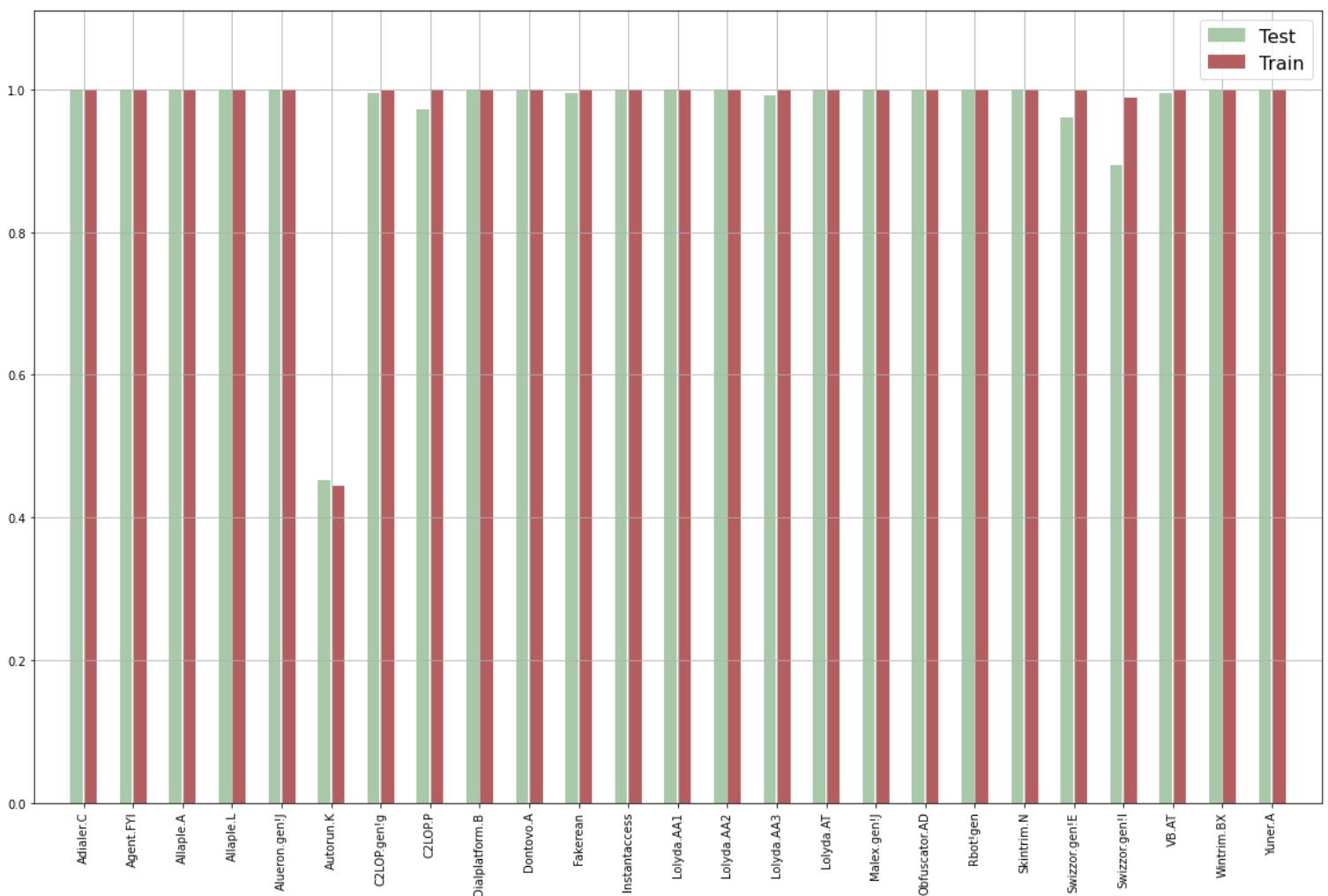
Success Rate (after normalizing):

- Train: 99.375 %
- Test: 99.109 %

Malware family with the highest false classification (after normalizing):

- Train: Autorun.K
- Test: Autorun.K





6 מסקנות

מחקר הפרויקט בא לבדוק האם שימוש במידה عمוקה באמצעות רשת CNN עדיף על פני שימוש בשיטות שונות מתחום למידת המכונה עבור בעית סיוג תמונות ובפרט, תמונות של נזקoot. בנוסף, הפרויקט בא לבצע השוואة למחקר [1].

6.1 מסקנות כלליות עבור מודלים פרימיטיביים

6.1.1 LDA

מודל זה נתן את התוצאות הטובות ביותר מבין כל המודלים הפרימיטיביים עבור גDAL התמונות $32 \times 32 - 1 \times 64$, תוצאות של $C-009-97.009$ ו- $C-97.347$ אחוזי הצלחה בהתאם. אם כי כאשר רץ המודל על קלט של 128×128 , ניתן היה להבחין בירידה דרסטית באחיזי הצלחה – 70.7%. בנוסף, ניתן לראות שעבור גDAL התמונות $32 \times 32 - 1 \times 64$ מירב השגיאות של המודל בסיווג הנאקות אירעו עבור שתי משפחות עיקריות ($I!E$, $Swizzor.gen!E$, $Swizzor.gen!I$), אם נבחן את תוצאות מודלים אלו ללא התייחסות למושכות אלה, נוכל הגיעו לאחיזי דיקוטובים בהרבה. לעומת זאת, כאשר אנו משתמשים על המודול 128×128 , ניתן לראות בעיה רוחבית בתהיליך הסיווג – לא רק עבור שתי המשפחות שצינו לעיל, אלא עבור מספר רב של משפחות. נוכל לפרק בעיה זו לשני חלקים. החלק הראשון של בעיה זו הוא בעית סיוג התמונות עבור כלל גDAL התמונות. חלקה השני של הבעיה מתרחש רק עבור גודל התמונות 128×128 , חלק זה מעלה בעיה רוחנית נוספת כיוון שהוא מוגדר רק מושכות אשר למודול יש קושי להוותנו וניתן לומר כי בעיה זו נגרמת כתוצאה ממספר מאפיינים גדול יחסית למספר התמונות הנבדקות – על בעיה זו נרחב בהמשך.

היבט נוסף עליו רצוי לתת את הדעת הוא זמני ריצת המודול עבור כל אחד מהגדלים השונים. עבור אלגוריתם זה ניתן לראות זמני ריצה איטיים יחסית בהשוואה לשאר המודלים. דבר אשר יכול להוות מכשול כאשר מעוניינים לפתור בעיה זו בסביבה דלת אמצעים.

6.1.2 QDA

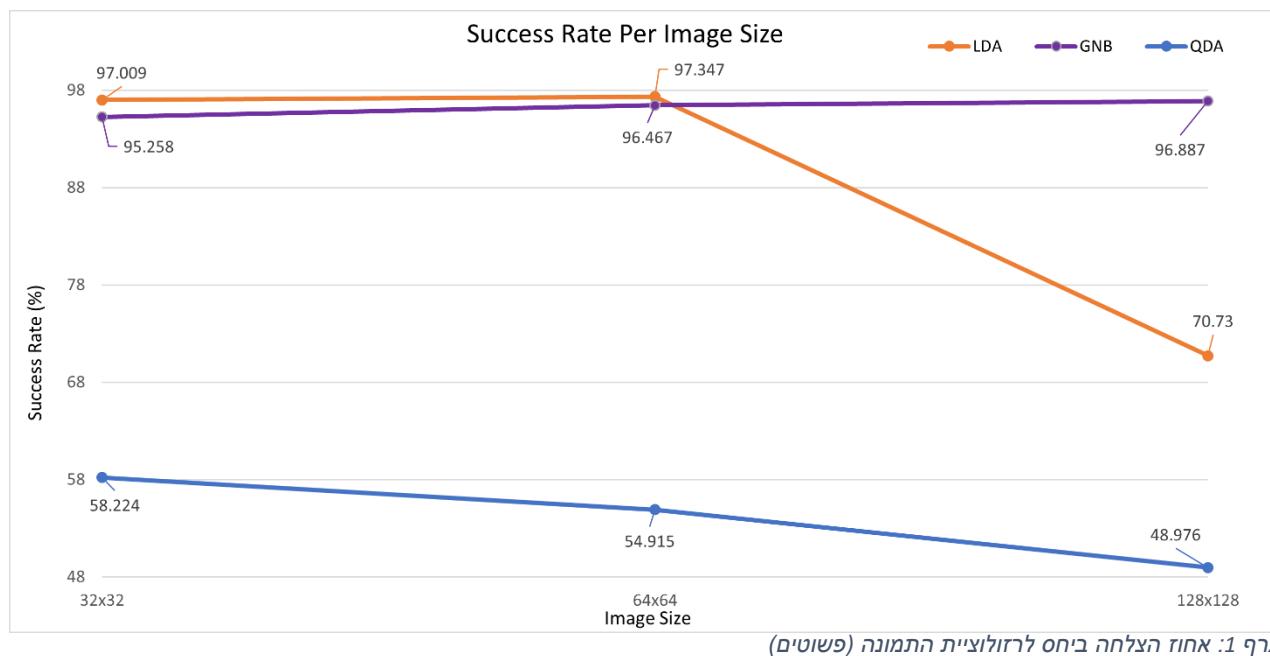
מודל זה נתן את התוצאות הגרועות ביותר מבין כל המודלים הפרימיטיביים עבור כלל גDAL התמונות. כבר בשלבי ריצת המודול ניתן היה לשים לב בעיה הנקראת Variable Collinearity אשר מתרחשת כתוצאה מקורלצייה חזקה מאוד בין שני משתנים שונים המקשת מאוד לביצוע הערקה/סיווג על כל אחד מהם בנפרד. בעקבות בעיה זו, ניתן להסיק שהמודול אינו מתאים לפתרון הבעיה בה עסק הפרויקט או לחילופין, מודל זה ידרש התאמות רבות בبنית המודול ועיצוב הנתונים. התאמות אלו אינן כוללות בסkop הלימודי של פרויקט זה ועל כן לא נעסק בכך.

6.1.3 GNB

מודל זה נתן את התוצאות היציבות ביותר ביותר מבין כל המודלים הפרימיטיביים עבור כל גDAL התמונות. ככלומר, בכל אחד מהמודלים הנבדקים התקבלו תוצאות גבוההות מאודם ויציבותם. כמו כן, מודל זה הציג זמני ריצה טובים מאוד ביחס לכל המודלים שנבדקו בפרויקט, דבר שיש לקחת בחשבון בעת ניתוח והבנת יתרונותיהם וחסרונותיהם של האלגוריתם וסביבה העבודה. אנו סבורים כי בעית הממדים ("Curse of Dimensionality") לא קורת במודל זה על אף שכמה הממדים גדולות באופן יחסי, בנוסף לכך שכמה הממדים זהה בכל המודלים. זאת בגיןו לכך שבצורה כזו או אחרת בעיה זו אירעה בכל אחד מן המודלים הפרימיטיביים האחרים. ניתן להסיק כי עבור מודל זה בעיה זו לא מתקיימת וניתן לראות תוצאות יציבות מכיוון שלמודל זה מאפיינים והנחות אשר מסוימים לכך. מודל זה בעל הנחיה מרכזית מושגת לפחות תלות סטטיסטיות בין כל מאפייני (Features) המודל אינה קיימת. עם זאת, על אף יתרונותיו הרבים של אלגוריתם זה, קיימים חסרים מרכזי להנחותיו. חיסרונו זה הוא יכולת מיפוי ויזומי דפוסים מוגבלת הן במספרן והן במורכבותן. לכן, לא ניתן לומר באופן חד משמעי כי מודל זה מסוגל להתגבר על בעיה זו גם כאשר כמה מושפחות (Classes) הנתונות למודל תגדל (בשילוב עם כמה הדוגמאות). כדיו כמה הדוגמאות עבור המודל שלנו מוגבלות ומספרן

באופן ייחסי נמוך. לפיכך, במידה ועליה הצורך לפטור את בעיית הסיווג הנתונה בפרויקט זה עם כמות מושיפות רחבה יותר, ניתן כי מודל שכזה לא יכול לחתמווד עם מודלים מורכבים יותר כגון מודלי רשותות עצביות, אלא אם כן, מספר הדוגמאות יגדל משמעותית ביחס למספר המשפחות הגדל (אם מספר הדוגמאות ישאר נמוך עדין נוכל להמשיך ולראות תוצאות יחסית טובות גם בהשוואה למודלי רשותות עצביות).

תופעה מעניינת אשר ניתן לראות במודל זה היא שעבור רזולוציות של 64×64 ו- 128×128 , ה-K שנותן את התוצאות הטובות ביותר היה $3 = K$. ככלומר, המודל אומן על $\frac{2}{3}$ מהדוגמאות ונבדק על $\frac{1}{3}$ מהן. דבר זה מעיד על מודל חזק שבו ניתן עם מעט דוגמאות להציג ליכולת דיקג בוגה וכל זאת כאשר כמות הדוגמאות עבור סדרת המבחן גדולה.



גרף 1: אחוז הצלחה ביחס לרזולוציות התמונה (פשוטים)

6.2 מסקנות עבור מודלי CNN

6.2.1 מסקנות כלליות

ניתן לומר באופן גורף כי כל המודלים הנבדקים בפרויקט זה מסוג CNN נתנו תוצאות וביצועים טובים מאוד בנוסף ליציבותם בגובהה. זאת מכיוון שאלגוריתם זה בסיסו בעל יכולות זיהוי ומיפוי דפוסים, בדגש על דפוסים בעת סיווג תമונות.

יכולת זיהוי ומיפוי דפוסים זאת נובעת בעיקר מruntime של אלגוריתם ה-CNN. ניתן לומר כי יכולות זיהוי היכולת המרכזית אשר יוצרת את ההבדל בין אלגוריתם זה לבין שאר האלגוריתמים. תהליך מיפוי זיהוי הדפוסים מתרחש בשכבות הקונבולוציה ובו קיים ניסיון לחקוק את האופן שבו בני האדם תופסים את הסביבה באמצעות העיניים, אשר בעת תהליך האיזוי העין האנושית מחלקת את התמונה הרחבה לתת תמונות קטנות ומנתחת (באמצעות המוח) אותן אחת אחת. לשם כך בתהליך הממחקר נבדקו מגוון רחב של פילטרים אשר אנו טוענים כי השפיעו על תהליכי הקונבולוציה בראשת אופן משמעותי ותרמו באופן ניכר לתוצאותיו של המודל. עוד בנוגע ליתרונותיו של אלגוריתם זה הם

שכבות ה-Pooling וה-Fully Connected Layer אשר בהן אנו משתמשים בחלוקת המודלים. Pooling Layer מעניקה את יכולת הסינון והשימור של מאפיינים קטנים תוך הפקת מספר הפיצרים אותם לבסוף נספיק לשכבה ה-Fully Connected. זו תעניק למודל את יכולת הסיווג וחיה הקשרים. בנוסף, אנו סבורים כי אחת הסיבות אשר העניקה את יכולות אלו היא היכולת של מודלים מסוג CNN לבחור באופן עצמאי את הפיצרים הרלוונטיים עבורם ללא התערבות האדם ואת בנגוד לשאר המודלים

אשר נאלצו לבצע את תהליכי הסיווג בעזרת כלל הפעיצרים (פיקסלים) בלבד לבצע סינון מתאים. מאידך, במקרה המדובר בפרויקט זה, שכבת הקלט הורכבה מפיקסלים אשר בנויים מצלע אחד ולא שלושה צבעים (כלומר, מטריצה אחת, דו-ממדית). במקרה מסוים, דבר זה פישט את הבעיה ותרם לפתרונה ברמת הביצועים ומני הריצה. דבר שוביל במרקם אחרים לא היה מתקיים.

6.2.2 מסקנות ביחס למודלים

כמתואר בגרף (2), ניתן לראות את אחוזי ההצלחה של מודלי ה-CNN ביחס לגודלי התמונות השונים. כמו כן, אפשר לשים לב כי עבור כל המודלים (באופן כללי) קיימת מגמת שיפור כתלות במספר הפעיצרים. ניתן לראות כי המודל המורכב (Complex model) הפיק תוצאות פחותות מיתר המודלים אשר על אף מורכבותו (עוד מגוון שכבות) אשר קיוינו שתספק תוצאות טובים יותר, וזאת על אף שמנוי הריצה שלו הם הגבוהים ביותר מבין מודלי CNN. על כן, מצאנו כי מודל בעל מבנה שכזה אינו כדאי הן מבחינת הביצועים והן מבחינת זמני הריצה. בכל אחת מהרולוציות השונות, ניתן לשים לב לקורלציה מסוימת בין המודלים (בהתאמה זו נתעלם מהמודל המורכב).

6.2.2.1 32×32

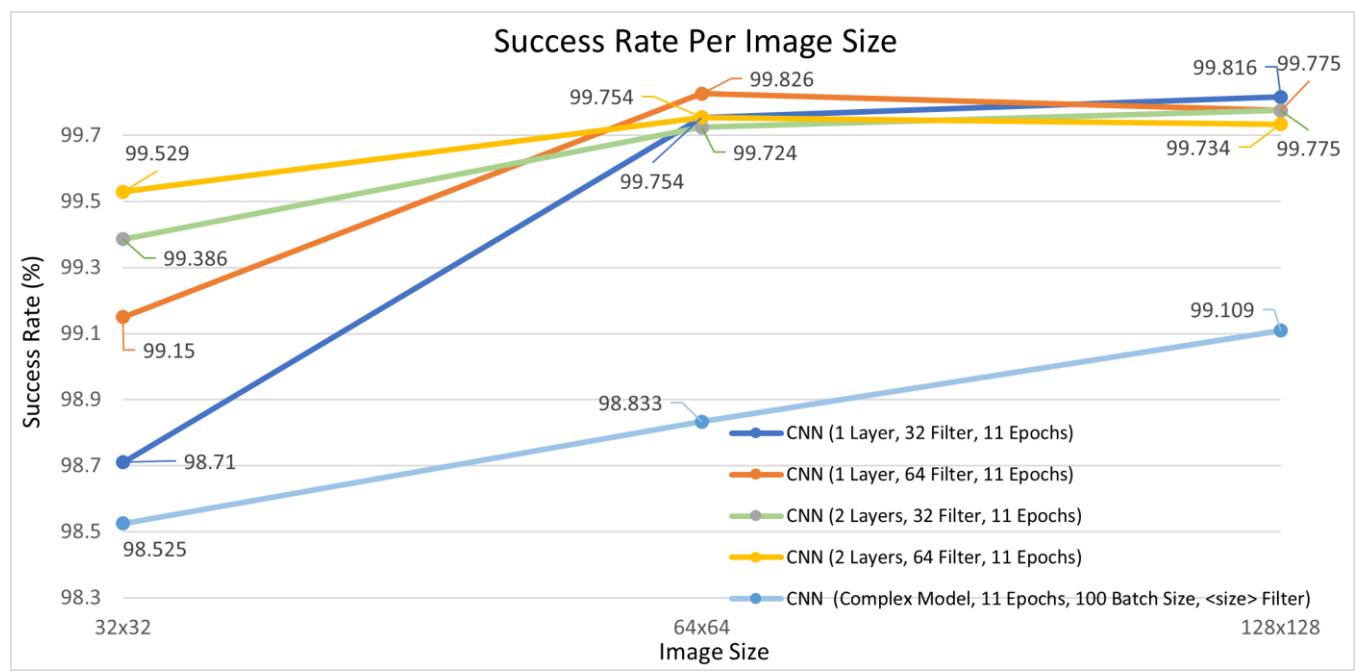
ברגולוציה זו ניתן לראות פיזור על פני טווח ערכים רחב (~1%). בנוסף, ניתן לראות כי עבור ארבעת המודלים ה-"פостиים", המודל בעל 64 פילטרים מספק באופן עקבי תוצאות טובות יותר מזו של המודלים בעלי 32 פילטרים (שכפת אחת, שתי שכבות).

6.2.2.2 128×128

ברגולוציות אלו, ניתן לראות מקבץ המתפזר על פני טווח ערכים קטן מאוד. לכן, במקרה זה לא נוכל לקבוע כי קיימת מגמה/סדר מסויים בין המודלים השונים.

נקודות נס포ת:

- עבור התוצאות עליהן אנו מתבוננים, המודל בעל שכבה אחת ו-64 פילטרים הוא המודל אשר נתן את התוצאה הטובה ביותר והוא 99.826 אחוזי ההצלחה.
- המודל המשורטט בצבע כחול, בעל שכבה אחת ו-32 פילטרים, הוא מודל בעל מגמת השיפור הגדולה ביותר מבין כל המודלי CNN, מגמת שיפור של 1.675%.



graf 2 : אחוז ההצלחה ביחס לרגולוציות התמונה (CNN)

6.2.3 השוואת ביחס למחקר

ביחס למחקר בתחום, עליו ביססנו את הפרויקט, ניתן לראות כי הצלחנו לשפר את כלל המודלים שנבדקו במספר אchosim. למשל:

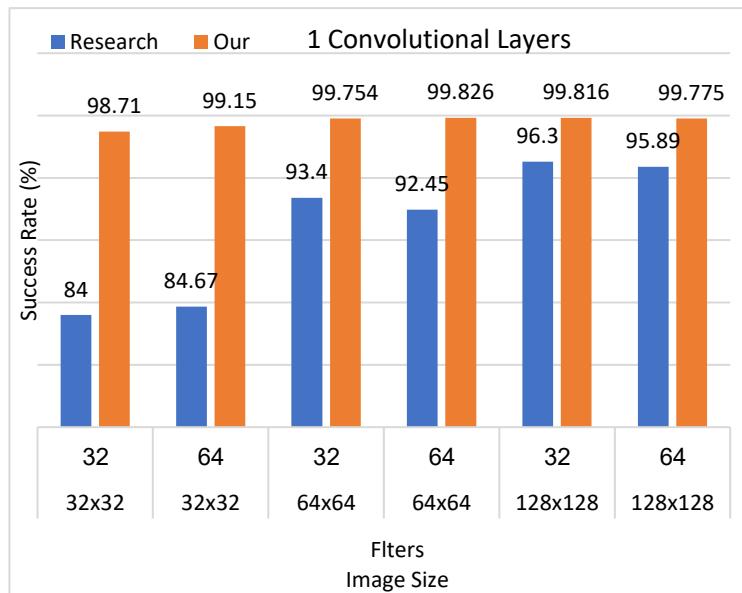
- עבור רשת CNN 128×128 , בעלת 2 שכבות קונבולוציה ו-64 פילטרים, הממחקר השיג כ-95.7%

אחווי הצלחה (המודול בעל הביצועים הטובים ביותר במחקר). אילו מנגד, אנו, בפרויקט זה, הצלחנו להשיג כ-99.7% אחווי הצלחה.

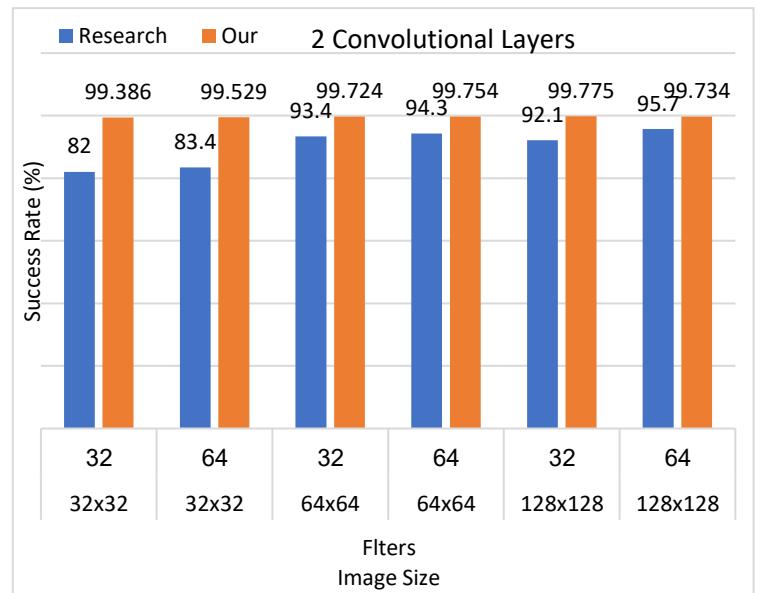
- מהתובנות על המודלים המוצלחים ביותר במחקר ניתן לראות כי המודלים מספקים 0

אחווי הצלחה עבור סיווג משפחת הנזקוקות K. Autorun. לעומת זאת, בפרויקט זה, עבור המודלים המקבילים למודלים אלו, הצלחנו לשפר את פרמטר זה ואף להגיע ל-100% אחווי הצלחה, לדוגמא במודול 64×64 שכבה 1 ו-64 פילטרים כמתואר בגרף (3, 4).

כאמור, כתבי הממחקר מפרטים רק על העקרונות שעלייהם התבססו בתהיליר בנויות המודלים. כתוצאה לכך, נעזרנו בעקרונות אלה כקווים מנחים על מנת לנסות ולהתחקות לאופן פעולתם.



גרף 4: השוואת ביחס למחקר, שכבה 1



גרף 4: השוואת ביחס למחקר, 2 שכבות

בגרפים אלו ניתן לראות השוואת אחווי הצלחה של מודלי הממחקר לבין מודלי הפרויקט שלנו.

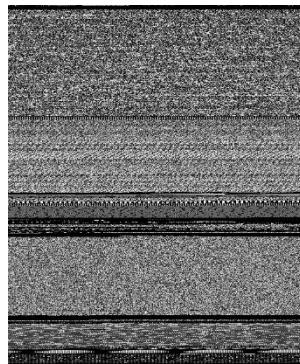
6.3 משפחות בעלות דמיון

על סמך התוצאות (פרק 6) וניתוח הנתונים (פרק זה) ניתן להזיהות סוגים של משפחות אשר בהן קיימת בעיה בסיווגן בשל דמיון בין המשפחות במספר אופנים.

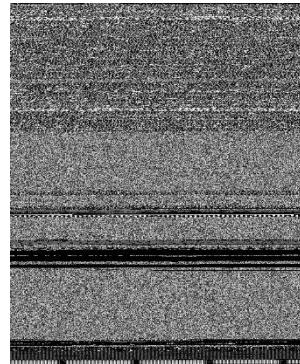
6.3.1 משפחות בעלות דמיון – כללי

בחלק זה נתאר את הבעיה עבור משפחות אשר ניתן לראות את הקושי בסיווגן בכלל המודלים.

- משפחתי! Swizzor.gen!, הכלולת בתוכה את E! Swizzor.gen!-I!, אשר היו בעיה בסיווגן עבור כלל המודלים. דבר זה לא מפתיע, זאת מכיוון כי מדובר בשני וריאנטים של אותה המשפחה.



Swizzor.gen!I

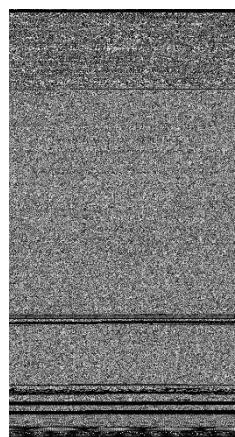


Swizzor.gen!E

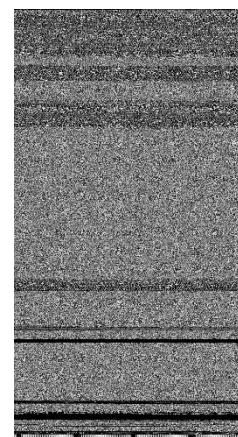
איור 13: משפחתי! Swizzor.gen!

בנוסף, ניתן לראות את הדמיון הרב בין שתי התמונות כפי שמצויר באיור (14).

- משפחתי C2LOP, הכלולת בתוכה את !C2LOP.gen!C2LOP.P, אשר היו בעיה בסיווגן עבור כלל המודלים. דבר זה לא מפתיע, זאת מכיוון כי מדובר בשני וריאנטים של אותה המשפחה. בעיית הסיווג אותה זיהינו עבור משפחה זו היא ש-P. C2LOP מזדהה כמו .C2LOP.gen!g



C2LOP.P



C2LOP.gen!g

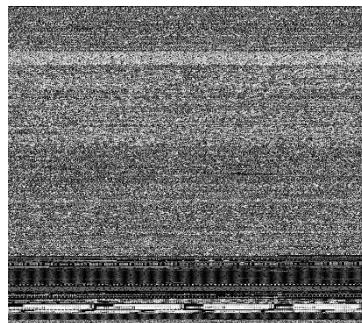
איור 14: משפחתי! C2LOP

בנוסף, ניתן לראות את הדמיון הרב בין שתי התמונות כפי שמצויר באיור (15).

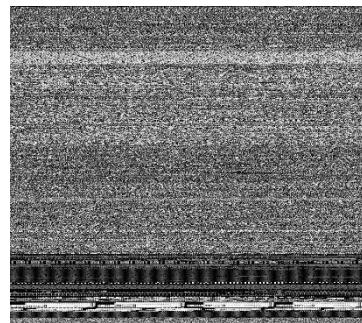
6.3.2 משפחות בעלות דמיון – מודלים ספציפיים
בחלק זה נתאר את הבעיה עבור משפחות בעלות דמיון אשר ניתן לראות את הקשיי בסיווג
בהתיחסות למודלים ספציפיים.

▪ CNN

- עבור אלגוריתם זה זיהינו שתי משפחות אשר קיימת בעיה בסיווג ברוב המודלים ואילו
ביתר המודלים מקבלים יותר פיצרים (אלו מודלים בעלי אחווי הצלחה גבוהה יותר),
בעיה זו נפתרה (אצלנו). הבעיה אליה נחשפנו היא שתמונות אשר שייכות למשפחה
זוהו כחלק ממינימום Autorun.K .Yunar.A



Autorun.K



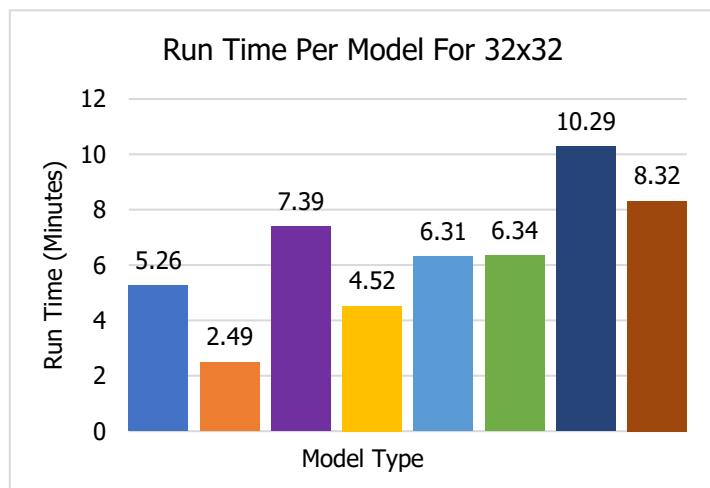
Yunar.A

איור 15: משפחות A & Yunar.K

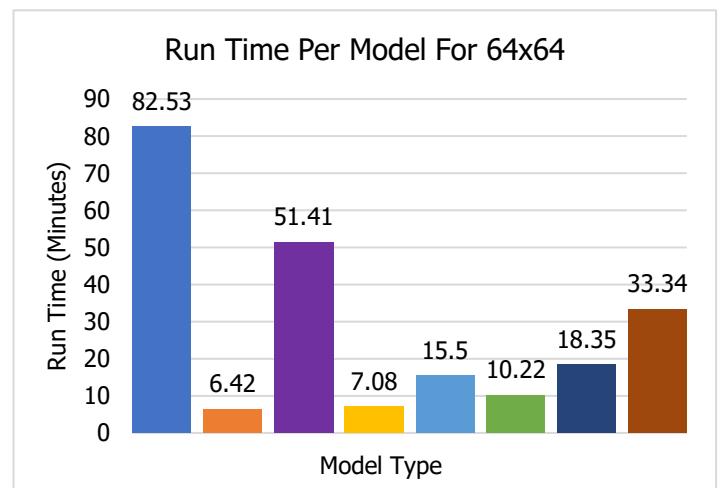
חומר היכולת בסיווג משפחות אלה היה מפתיע ואת מכיוון שמדובר בשתי משפחות שונות
לחלוטין האחת מהשנייה. אך כפי שניתן לראות באיזור (16), קיים דמיון רב בין שתי
משפחות אלה.

6.4

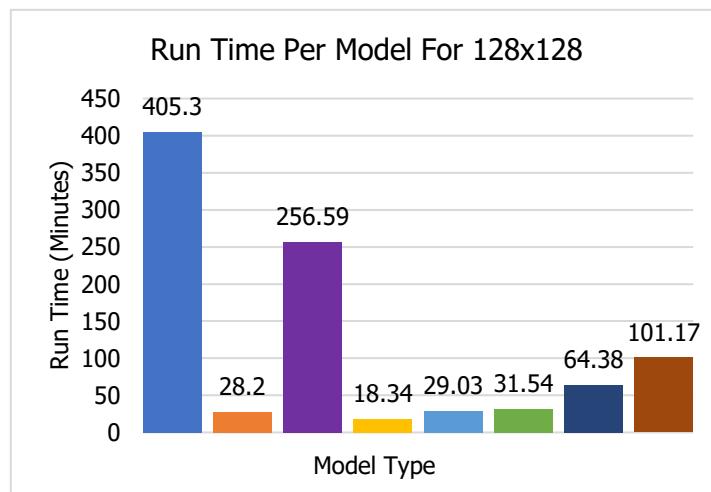
בעת ניתוח נושא זה, ניתן להבחן ב-2 מגמות הנקוטות. האחת, ב- 32×32 פיקסלים והיא מגמה אשר למודלים הפרימיטיביים יש זמני ריצה נמוכים יחסית ביחס למודלי CNN ובמגמה השנייה שמתקיימת עבור 64×64 ו- 128×128 פיקסלים, ניתן להבחין כי זמני הריצה של מודלי CNN משתפרים משמעותית ביחס למוגני הריצה של המודלים הפרימיטיביים. חשוב לזכור כי חלק מסווקלי בהערכת המודל בהתחשב במשאיבי סבירות העבודה אשר עליה המודל יrotch. כאמור, כאשר קיימת דרישת לביצועים מסווקים בשילוב לשביבת עבודה דלת אמצעים מודל 32×32 GNB אשר בעל אחווי הצלחה ומוגני ריצה יחסית יכול להשתאים, אך במקרים אחרים בהם נדרש אחווי הצלחה טובים יותר כנראה שמודל שזכה לאיספיק. כאשר אנו מתבוננים על מודלים בעלי רזולוציה של 64×64 ומעלה, ניתן לראות בוירור את חוסר הcadיות בשימוש במודלים הפרימיטיביים אל מול מודלי CNN, ואאת בשל אחווי ריצה גבוהים ואחווי הצלחה נמוכים בגין מודלי CNN אשר בעל זמני ריצה טובים ואחווי הצלחה גבוהים מאוד. גם כאשר אנו מתבוננים רק על מודלי CNN ניתן להבחין בחוסר כדיות של המודל המורכב (Complex Model) וזאת מכיוון שזמן ריצתו של מודל זה גדול משמעותית מזמני הריצה של המודלי CNN האחרים על אחת כמה וכמה שאחווי הצלחתו נמוכים/זהים למודלים האחרים.



גרף 7 : זמני ריצה 32×32



גרף 7 : זמני ריצה 64×64



גרף 7 : זמני ריצה 128×128

- LDA
- QDA
- CNN (1 Layer, 64 Filter, 11 Epochs)
- CNN (2 Layers, 64 Filter, 11 Epochs)
- GNB
- CNN (1 Layer, 32 Filter, 11 Epochs)
- CNN (2 Layers, 32 Filter, 11 Epochs)
- CNN (Complex Model, 11 Epochs, 100 Batch Size, <size> Filter)

הפרויקט חקר את מאגר הנתונים Malimg בשלוש וריאציות שונות של גודלי תמונות: 32×32 , 64×64 , 128×128 , כאשר כל וריאציה הייתה קלט עבור כל אחד מהמודלים שנבדקו (CNN, QDA, LDA, GNB). על סמך התוצאות שקיבלנו, בבדיקה סדרת המבחן עבור כל אחד מהמודלים, ניתן להסיק כי כל אחת מרשותות ה-CNN שהופעלו השיגה תוצאה טובה יותר מיתר המודלים (QDA, LDA, GNB) עבור אותה הקטגוריה. בנוסף, גם ביחס למאמר בתחום, עליו ביססנו את הפרויקט, ניתן לראות כי הצלחנו לשפר את כלל המודלים שנבדקו במספר אchosim. למשל, עבור 128×128 , 2 Convolutional Layers, 64 Filters CNN המבחן במאמר השיג כ-95.7% הצלחה ואילו מנגד, הצלחנו להשיג כ-99.7% אchosim הצלחה. כאמור, כתוביו המאמר מפרטים רק על העקרונות שעל בסיסם בנו את המודלים בניסוי, וכ吐וצאה מכך, נעזרנו בעקרונות אלה כקווים מנחים על מנת לנסות ולהתחקות לאופן פועלתם. באופן כללי, ניתן לומר כי מודלי ה-CNN סייפקו את אchosim הדיווק הטוביים ביותר.

באשר למודלי למידת המכונה בהם השתמשנו, ניתן לראות בתוצאות שקיבלנו עבור A-LDA, בקטגוריות של 64×32 & 32×64 , השגנו תוצאות גבוהות של כ-97% עבור כל אחד. אם כי כאשר רץ המודל על קלט של 128×128 , ניתן היה לבדוק בירידה דרסטיבית באchosim הצלחה – 70.7% בלבד. במודל ה-QDA ניתן לבדוק כי ככל הקטגוריות נכשלו מבחינת אchosim הצלחה כאשר כולם נמצאו מתחת ל-.60%.

אנו מסיקים כי ירידה כה דרסטיבית באchosim הצלחה עבור כל המקרים הללו נובעת מהגדלה הנקראית "Curse of Dimensionality" הנגרמת כתוצאה ממספר דוגמאות נמוך ביחס למספר קלטי המודל. עבור מודל ה-GNB, הצלחנו להשיג אchosim הצלחה גבוהה יחסית, כ-96% עבור כל אחת מהקטגוריות. אנו סוברים כי בעקבות ההנחה של אי תלות סטטיסטית אשר מניח אלגוריתם ה-GNB, הוא מצליח להתגבר על הבעה של קודמו ואט מכיוון שאינו הtalות מפחיתה את כמות המקרים שעל המודל ללמידה.

7 מקורות

- [1] M. Jain, "Image-Based Malware Classification with Convolutional Neural Networks and Extreme Learning Machines ",*SJSU ScholarWorks*, p. 67, 2019 .
- [2] L. Chen, R. Sahita, J. Parikh ו M. Marino, "STAMINA: Scalable Deep Learning Approach for Malware Classification," Intel, 2020.