UNIVERSITÀ DEGLI STUDI DI MESSINA

DIPARTIMENTO DI SCIENZE MATEMATICHE E INFORMATICHE,
SCIENZE FISICHE E SCIENZE DELLA TERRA

Corso di Laurea Magistrale in Matematica Applicata

Una classe Python per la Morfologia Neutrosofica

Tesi di Laurea di
Lorenzo AFFÉ

Relatore
Ch.mo Prof. Giorgio NORDO

ANNO ACCADEMICO 2022/2023

A mio fratello Riccardo.
Grazie al tuo coraggio sono riuscito a trovare il mio

Indice

Elenco delle figure

Capitolo 1

Introduzione

Nella società contemporanea, caratterizzata da una crescente complessità e interconnessione, le tradizionali teorie matematiche si trovano spesso a confrontarsi con limiti significativi, specialmente quando vengono applicate a scenari reali dove l'ambiguità e l'incertezza regnano sovrane. In particolare, la crescente necessità di analizzare e interpretare grandi volumi di dati visivi ha messo in luce la necessità di approcci matematici che possano gestire efficacemente le informazioni complesse e spesso ambigue contenute nelle immagini.

Nell'ambito della matematica applicata, il trattamento dell'incertezza e dell'ambiguità dei dati ha assunto un'importanza crescente, soprattutto in contesti dove le informazioni sono incomplete, indeterminate o contraddittorie. Questo è particolarmente evidente nel campo dell'analisi delle immagini, dove la capacità di interpretare accuratamente dati visivi può avere un impatto significativo in una vasta gamma di applicazioni, dalla diagnostica medica alla sorveglianza della sicurezza.

Questa tesi si propone di indagare due filoni di ricerca particolarmente promettenti per affrontare tali sfide: la Teoria Neutrosofica e la Morfologia Matematica. Entrambi gli approcci offrono nuove prospettive e strumenti per l'analisi e l'elaborazione delle immagini, un'area che richiede una particolare attenzione alla gestione dell'incertezza e dell'ambiguità.

La Teoria Neutrosofica, introdotta da Florentin Smarandache nei primi anni 2000, estende i concetti dei fuzzy set e gli intuitionistic fuzzy set per meglio rappresentare l'incertezza, l'indeterminatezza e la contraddizione all'interno di un unico quadro teorico. Attraverso questo approccio, la Teoria Neutrosofica risulta particolarmente utile nell'analisi delle immagini, dove le informazioni spesso non sono né completamente vere né completamente false, ma piuttosto esistono in uno stato di fluidità e ambiguità. Questa nuova logica matematica si adatta meglio alla complessità e alla natura sfumata del mondo reale, specialmente nell'interpretazione delle immagini, dove le sfumature di grigio e le forme indistinte richiedono un'analisi che vada oltre il binario e il deterministico.

Parallelamente, la Morfologia Matematica, quale branca della matematica applicata nata negli anni '60 dagli studi di Georges Matheron e Jean Serra, che si concentra sull'analisi e l'elaborazione delle immagini, si è rivelata essere uno strumento fondamentale che ha trovato applicazione in una varietà di campi che vanno dalla visione artificiale alla biologia computazionale.

La fusione di queste due discipline ha dato vita alla Morfologia Neutrosofica, una nuova frontiera che promette di rivoluzionare il trattamento e l'analisi delle immagini in condizioni di incertezza elevata.

In questa tesi, esploriamo la profondità e l'applicabilità della Morfologia Neutrosofica, dimostrando come essa non solo superi le limitazioni dei tradizionali approcci matematici

ma offra anche una metodologia più robusta e versatile per affrontare problemi complessi in vari campi scientifici e ingegneristici.

L'obiettivo di questa ricerca è duplice: da un lato, mira a fornire una comprensione approfondita dei principi e delle applicazioni della Morfologia Neutrosofica, dall'altro, cerca di evidenziare il suo potenziale come strumento innovativo nell'elaborazione delle immagini, soprattutto in casi con un elevato grado incertezza e nel farlo ci focalizziamo sulla pratica applicazione di questi concetti teorici, come dimostrato dall'implementazione di una nuova classe Python, NSmorph, descritta nell'ultimo capitolo della tesi.

La classe NSmorph rappresenta un'innovazione significativa nel campo dell'elaborazione delle immagini neutrosofiche, posizionandosi nel nascente settore della Morfologia Neutrosofica Computazionale. Sviluppata in Python, questa classe utilizza librerie standard come NumPy, MatplotLib e OpenCV-Python per manipolare immagini secondo i principi della Morfologia Neutrosofica. Attraverso una serie di metodi definiti all'interno della classe, come dilatazione, erosione, apertura e chiusura neutrosofica, NSmorph consente di applicare operazioni morfologiche neutrosofiche in modo intuitivo ed efficace. L'uso di Python, con la sua sintassi chiara e la sua vasta gamma di librerie scientifiche e matematiche, rende la classe NSmorph particolarmente accessibile e potente. Questo facilita non solo lo sviluppo e il test di nuove tecniche di elaborazione delle immagini, ma anche la loro applicazione pratica a problemi reali, come dimostrato da diversi esempi e casi di studio presentati nella tesi.

In definitiva, questa tesi non solo approfondisce la comprensione teorica della Morfologia Neutrosofica ma estende anche le sue applicazioni pratiche, aprendo nuove strade per la ricerca futura e per lo sviluppo di strumenti di elaborazione delle immagini più sofisticati e sensibili al contesto. Attraverso l'implementazione della classe NSmorph, dimostriamo come i concetti avanzati di morfologia neutrosofica possano essere tradotti in strumenti pratici, fornendo un contributo significativo sia al campo accademico che a settori specificatamente applicativi.

Capitolo 2

Insiemi neutrosofici

2.1 Introduzione

La nozione di *insieme neutrosofico* fu introdotta nel 1999 da Smarandache [16] come una generalizzazione delle teorie dei *fuzzy sets* introdotti da Zadeh nel 1965 [19] e degli *intuitionistic fuzzy sets* introdotti da Atanassov nel 1983 [1]. Nella logica neutrosofica, una proposizione ha un grado di verità, un grado di indeterminatezza ed un grado di falsità.

Notazione 1. *Sia $\mathbb{U}$ un insieme, $I = [0,1]$ l'intervallo unitario dei numeri reali, per ogni $r \in I$, con $\underline{r}$ denotiamo la funzione costante $\underline{r} : \mathbb{U} \to I$ tale che, per ogni $u \in \mathbb{U}$ è definita da $\underline{r}(u) = r$.*
Per ogni famiglia $\{f_i\}_{i\in I}$ di funzioni $f_i : \mathbb{U} \to I$, denotiamo con:

- *$\bigwedge_{i\in I} f_i$ la funzione estremo inferiore $\bigwedge_{i\in I} f_i : \mathbb{U} \to I$ tale che, per ogni $u \in \mathbb{U}$ è definita da $\left(\bigwedge_{i\in I} f_i\right)(u) = \bigwedge_{i\in I} f_i(u) = \inf\{f_i(u) : i \in I\}$.*
- *$\bigvee_{i\in I} f_i$ la funzione estremo superiore $\bigvee_{i\in I} f_i : \mathbb{U} \to I$ tale che, per ogni $u \in \mathbb{U}$ è definita da $\left(\bigvee_{i\in I} f_i\right)(u) = \bigvee_{i\in I} f_i(u) = \sup\{f_i(u) : i \in I\}$.*

In particolare, se f e g sono due funzioni da $\mathbb{U}$ a I, denotiamo il loro estremo inferiore (che è il minimo) con $f \wedge g$ ed il loro estremo superiore (che è il massimo) con $f \vee g$.

Definizione 2.1.1. [17] *Sia $\mathbb{U}$ un insieme universo iniziale e $A \subseteq \mathbb{U}$, un **single valued neutrosophic set** su $\mathbb{U}$ (**SVN-set** in breve), dentotato con $\widetilde{A} = \langle \mathbb{U}, \mu_A, \sigma_A, \omega_A \rangle$, è un insieme della forma*

$$\widetilde{A} = \{(u, \mu_A(u), \sigma_A(u), \omega_A(u)) : u \in \mathbb{U}\}$$

*dove $\mu_A : \mathbb{U} \to I$, $\sigma_A : \mathbb{U} \to I$ e $\omega_A : \mathbb{U} \to I$ sono rispettivamente la **funzione di appartenenza**, la **funzione di indeterminatezza** e la **funzione di non appartenenza** di A. Per ogni $u \in \mathbb{U}$, $\mu_A(u)$, $\sigma_A(u)$ e $\omega_A(u)$ vengono chiamati rispettivamente **grado di appartenenza**, **grado di indeterminatezza** e **grado di non appartenenza** di u.*

Considerando che $I = [0,1]$, chiaramente risulta che $0 \leq \mu_A(u) + \sigma_A(u) + \omega_A(u) \leq 3$, per ogni $u \in \mathbb{U}$.

Notazione 2. *L'insieme di tutti i single valued neutrosophic sets su un insieme universo $\mathbb{U}$ sarà denotato con $\mathcal{SVN}(\mathbb{U})$.*

Definizione 2.1.2. [16, 17] *Siano $\widetilde{A} = \langle \mathbb{U}, \mu_A, \sigma_A, \omega_A \rangle$ e $\widetilde{B} = \langle \mathbb{U}, \mu_B, \sigma_B, \omega_B \rangle$ due SVN-set sullo stesso insieme universo $\mathbb{U}$, diciamo che $\widetilde{A}$ è un **sottoinsieme neutrosofico** (o semplicemente un sottoinsieme) di $\widetilde{B}$ e scriviamo $\widetilde{A} \Subset \widetilde{B}$ se, per ogni $u \in \mathbb{U}$, risulta $\mu_A(u) \leq \mu_B(u)$, $\sigma_A(u) \leq \sigma_B(u)$ e $\omega_A(u) \geq \omega_B(u)$. Diremo anche che $\widetilde{A}$ è contenuto in $\widetilde{B}$ o che $\widetilde{B}$ contiene $\widetilde{A}$.*

È interessante notare che la relazione $\Subseteq$ soddisfa le proprietà riflessiva, antisimmetrica e transitiva e quindi $(\mathcal{SVN}(\mathbb{U}), \Subseteq)$ forma un insieme parzialmente ordinato o poset (dall'inglese *partially ordered set*), ma non un insieme totalmente ordinato o loset (dall'inglese *linearly ordered set*) come mostrato nel seguente esempio in cui si evidenzia che in $\mathcal{SVN}(\mathbb{U})$ possono esistere elementi non confrontabili.

Esempio 2.1.3. *Siano $\mathbb{U} = \{a, b, c\}$ un insieme universo finito, $\widetilde{A} = \langle \mathbb{U}, \mu_A, \sigma_A, \omega_A \rangle$, $\widetilde{B} = \langle \mathbb{U}, \mu_B, \sigma_B, \omega_B \rangle$ due SVN-set su $\mathcal{SVN}(\mathbb{U})$ definiti rispettivamente dalla seguente rappresentazione tabulare:*

$\mathbb{U}$ \ $\widetilde{A}$	μ_A	σ_A	ω_A
a	0.5	0.3	0.2
b	0.6	0.2	0.3
c	0.4	0.2	0.7

$\mathbb{U}$ \ $\widetilde{B}$	μ_B	σ_B	ω_B
a	0.2	0.2	0.2
b	0.4	0.1	0.6
c	0.8	0.3	0.1

Allora $\widetilde{A} \nsubseteq \widetilde{B}$ perché $\mu_A(a) = 0.5 > 0.2 = \mu_B(a)$ e $\widetilde{B} \nsubseteq \widetilde{A}$ perché $\omega_B(c) = 0.1 < 0.7 = \omega_A(c)$, dunque gli SVN-set $\widetilde{A}$ e $\widetilde{B}$ non sono confrontabili.

Definizione 2.1.4. [16, 17] *Siano $\widetilde{A} = \langle \mathbb{U}, \mu_A, \sigma_A, \omega_A \rangle$ e $\widetilde{B} = \langle \mathbb{U}, \mu_B, \sigma_B, \omega_B \rangle$ due SVN-set sull'insieme universo $\mathbb{U}$, diciamo che $\widetilde{A}$ è **neutrosoficamente uguale** (o semplicemente uguale) a $\widetilde{B}$ e scriviamo $\widetilde{A} \equiv \widetilde{B}$ se $\widetilde{A} \Subseteq \widetilde{B}$ e $\widetilde{B} \Subseteq \widetilde{A}$.*

Definizione 2.1.5. [17] *L'SVN-set $\langle \mathbb{U}, \underline{0}, \underline{0}, \underline{1} \rangle$ è detto **insieme neutrosofico vuoto** su $\mathbb{U}$ e viene indicato con $\widetilde{\emptyset}$, o più precisamente da $\widetilde{\emptyset}_{\mathbb{U}}$ in caso sia necessario specificare il corrispondente insieme universo.*

Definizione 2.1.6. [17] *L'SVN-set $\langle \mathbb{U}, \underline{1}, \underline{1}, \underline{0} \rangle$ è detto **insieme neutrosofico assoluto** su $\mathbb{U}$ e viene indicato con $\widetilde{\mathbb{U}}$.*

È evidente che, per ogni $\widetilde{A} \in \mathcal{SVN}(\mathbb{U})$, risulta $\widetilde{\emptyset} \Subseteq \widetilde{A} \Subseteq \widetilde{\mathbb{U}}$.

2.2 Operazioni neutrosofiche

Definizione 2.2.1. [12] *Sia $\widetilde{A} = \langle \mathbb{U}, \mu_A, \sigma_A, \omega_A \rangle$ un SVN-set sull'insieme universo $\mathbb{U}$, il **complementare neutrosofico** (o, semplicemente, il complementare) di $\widetilde{A}$, denotato con $\widetilde{A}^{\complement}$, è il SVN-set $\widetilde{A}^{\complement} = \langle \mathbb{U}, \underline{1} - \mu_A, \underline{1} - \sigma_A, \underline{1} - \omega_A \rangle$.*

È semplice verificare che per ogni $\widetilde{A} = \langle \mathbb{U}, \mu_A, \sigma_A, \omega_A \rangle \in \mathcal{SVN}(\mathbb{U})$, risulta $\left(\widetilde{A}^{\complement}\right)^{\complement} \equiv \widetilde{A}$ e, in particolare, che $\widetilde{\mathbb{U}}^{\complement} \equiv \widetilde{\emptyset}$ e $\widetilde{\emptyset}^{\complement} \equiv \widetilde{\mathbb{U}}$.

Proposizione 2.2.2. [17] *Per ogni coppia $\widetilde{A} = \langle \mathbb{U}, \mu_A, \sigma_A, \omega_A \rangle$ e $\widetilde{B} = \langle \mathbb{U}, \mu_B, \sigma_B, \omega_B \rangle$ di SVN-set in $\mathcal{SVN}(\mathbb{U})$, si ha che $\widetilde{A} \Subseteq \widetilde{B}$ se $\widetilde{B}^{\complement} \Subseteq \widetilde{A}^{\complement}$.*

Definizione 2.2.3. [9] *Sia $\left\{\widetilde{A}_i\right\}_{i \in I}$ una famiglia di SVN-set $\widetilde{A}_i = \langle \mathbb{U}, \mu_{A_i}, \sigma_{A_i}, \omega_{A_i} \rangle$ su uno stesso insieme universo $\mathbb{U}$, definiamo l'**unione neutrosofica** (o semplicemente l'unione della famiglia $\{A_i\}_{i \in I}$), denotata con $\bigcup_{i \in I} \widetilde{A}_i$, l'insieme neutrosofico $\widetilde{A} = \langle \mathbb{U}, \mu_A, \sigma_A, \omega_A \rangle$ con $\mu_A = \sup_{i \in I} \mu_{A_i}$, $\sigma_A = \sup_{i \in I} \sigma_{A_i}$ e $\omega_A = \inf_{i \in I} \omega_{A_i}$. In particolare, l'unione neutrosofica di due singoli SVN-set $\widetilde{A} = \langle \mathbb{U}, \mu_A, \sigma_A, \omega_A \rangle$ e $\widetilde{B} = \langle \mathbb{U}, \mu_B, \sigma_B, \omega_B \rangle$, denotata con $\widetilde{A} \Cup \widetilde{B}$, è l'insieme neutrosofico definito da $\langle \mathbb{U}, \mu_A \vee \mu_B, \sigma_A \vee \sigma_B, \omega_A \wedge \omega_B \rangle$.*

Definizione 2.2.4. [9] *Sia* $\left\{\widetilde{A}_i\right\}_{i\in I}$ *una famiglia di SVN-set* $\widetilde{A}_i = \langle \mathbb{U}, \mu_{A_i}, \sigma_{A_i}, \omega_{A_i}\rangle$ *su uno stesso insieme universo* $\mathbb{U}$*, definiamo l'**intersezione neutrosofica** (o semplicemente intersezione della famiglia* $\{A_i\}_{i\in I}$*), denotata con* $\bigcap\!\!\!\!\bigcap_{i\in I} \widetilde{A}_i$*, l'insieme neutrosofico* $\widetilde{A} = \langle \mathbb{U}, \mu_A, \sigma_A, \omega_A\rangle$ *con* $\mu_A = \inf_{i\in I} \mu_{A_i}$*,* $\sigma_A = \inf_{i\in I} \sigma_{A_i}$ *e* $\omega_A = \sup_{i\in I} \omega_{A_i}$*.*
In particolare, l'intersezione neutrosofica di due singoli SVN-set $\widetilde{A} = \langle \mathbb{U}, \mu_A, \sigma_A, \omega_A\rangle$ *e* $\widetilde{B} = \langle \mathbb{U}, \mu_B, \sigma_B, \omega_B\rangle$*, denotata con* $\widetilde{A} \Cap \widetilde{B}$*, è l'insieme neutrosofico definito da* $\langle \mathbb{U}, \mu_A \wedge \mu_B, \sigma_A \wedge \sigma_B, \omega_A \vee \omega_B\rangle$*.*

Definizione 2.2.5. [17] *Siano* $\widetilde{A} = \langle \mathbb{U}, \mu_A, \sigma_A, \omega_A\rangle$ *e* $\widetilde{B} = \langle \mathbb{U}, \mu_B, \sigma_B, \omega_B\rangle$ *due SVN-set su* $\mathbb{U}$*, diciamo che* $\widetilde{A}$ *e* $\widetilde{B}$ *sono **neutrosoficamente disgiunti** se* $\widetilde{A} \Cap \widetilde{B} \equiv \widetilde{\emptyset}$*. Al contrario, se* $\widetilde{A} \Cap \widetilde{B} \not\equiv \widetilde{\emptyset}$ *diciamo che* $\widetilde{A}$ ***incontra neutrosoficamente*** $\widetilde{B}$ *(o che* $\widetilde{A}$ *e* $\widetilde{B}$ *si incontrano neutrosoficamente).*

Definizione 2.2.6. [9] *Siano* $\mathcal{A}, \mathcal{B} \subseteq \mathcal{SVN}(\mathbb{U})$ *due famiglie non vuote di SVN-sets su* $\mathbb{U}$*, diciamo che* $\mathcal{A}$ ***interseca neutrosoficamente*** $\mathcal{B}$ *(o che* $\mathcal{A}$ *e* $\mathcal{B}$ *si intersecano neutrosoficamente) se ogni elemento di* $\mathcal{A}$ *interseca neutrosoficamente ogni elemento di* $\mathcal{B}$*, cioè per ogni* $\widetilde{A} \in \mathcal{A}$ *ed ogni* $\widetilde{B} \in \mathcal{B}$ *risulta* $\widetilde{A} \Cap \widetilde{B} \not\equiv \widetilde{\emptyset}$*.*
In particolare, se $\widetilde{C} = \langle \mathbb{U}, \mu_C, \sigma_C, \omega_C\rangle$ *è un SVN-set su* $\mathbb{U}$ *che interseca neutrosoficamente ogni elemento della famiglia* $\mathcal{A}$*, diciamo che* $\widetilde{C}$ *interseca neutrosoficamente* $\mathcal{A}$*.*

Osservazione 2.2.7. *È importante osservare che, al contrario della teoria classica, l'intersezione neutrosofica di un SVN-set con il suo complementare non è sempre l'insieme neutrosofico vuoto, e che l'unione neutrosofica di un SVN-set con il suo complementare non è sempre l'insieme neutrosofico assoluto. Infatti, se consideriamo l'insieme universo* $\mathbb{U} = \{a, b\}$*, il SVN-set* $\widetilde{A}$ *su* $\mathcal{SVN}(\mathbb{U})$ *ed il suo complementare* $\widetilde{A}^{\mathbb{C}}$ *definiti dalle seguenti rappresentazioni tabulari:*

$\mathbb{U}$ \ $\widetilde{A}$	μ_A	σ_A	ω_A
a	*0.2*	*0.6*	*0.8*
b	*1*	*0.5*	*0*

$\mathbb{U}$ \ $\widetilde{A}^{\mathbb{C}}$	$1-\mu_A$	$1-\sigma_A$	$1-\omega_A$
a	*0.8*	*0.4*	*0.2*
b	*0*	*0.5*	*1*

possiamo verificare facilmente che l'intersezione neutrosofica, $\widetilde{A} \Cap \widetilde{A}^{\mathbb{C}}$ *e l'unione neutrosofica,* $\widetilde{A} \Cup \widetilde{A}^{\mathbb{C}}$ *sono date, rispettivamente, dalle seguenti rappresentazioni tabulari:*

$\mathbb{U}$ \ $\widetilde{A} \Cap \widetilde{A}^{\mathbb{C}}$	$\mu_{A\cap A^C}$	$\sigma_{A\cap A^C}$	$\omega_{A\cap A^C}$
a	*0.2*	*0.4*	*0.8*
b	*0*	*0.5*	*1*

e

$\widetilde{A} \Cup \widetilde{A}^{\complement}$ / $\mathbb{U}$	$\mu_{A\cup A^C}$	$\sigma_{A\cup A^C}$	$\omega_{A\cup A^C}$
a	*0.8*	*0.6*	*0.2*
b	*1*	*0.5*	*0*

e dunque $\widetilde{A} \Cap \widetilde{A}^{\complement} \not\equiv \widetilde{\emptyset}$ *e* $\widetilde{A} \Cup \widetilde{A}^{\complement} \not\equiv \widetilde{\mathbb{U}}$.

Gli operatori neutrosofici di unione, intersezione e complementare soddisfano alcune relazioni simili a quelle della teoria classica degli insiemi, che sono raggruppate nella seguente proposizione.

Proposizione 2.2.8. [17] *Per ogni SVN-set* $\widetilde{A} = \langle \mathbb{U}, \mu_A, \sigma_A, \omega_A \rangle \in \mathcal{SVN}(\mathbb{U})$*, si ha:*

(1) $\widetilde{A} \Cup \widetilde{A} \equiv \widetilde{A}$

(2) $\widetilde{A} \Cup \widetilde{\emptyset} \equiv \widetilde{A}$

(3) $\widetilde{A} \Cup \widetilde{\mathbb{U}} \equiv \widetilde{\mathbb{U}}$

(4) $\widetilde{A} \Cap \widetilde{A} \equiv \widetilde{A}$

(5) $\widetilde{A} \Cap \widetilde{\emptyset} \equiv \widetilde{\emptyset}$

(6) $\widetilde{A} \Cap \widetilde{\mathbb{U}} \equiv A$

Dimostrazione. (1) Infatti, $\forall u \in \mathbb{U}$ si ha che:

$$\begin{aligned} \mu_{A\cup A}(u) &= \mu_A(u) \vee \mu_A(u) \\ &= \mu_A(u) \end{aligned}$$

$$\begin{aligned} \sigma_{A\cup A}(u) &= \sigma_A(u) \vee \sigma_A(u) \\ &= \sigma_A(u) \end{aligned}$$

$$\begin{aligned} \omega_{A\cup A}(u) &= \omega_A(u) \wedge \omega_A(u) \\ &= \omega_A(u) \end{aligned}$$

e dunque che $\widetilde{A} \Cup \widetilde{A} \equiv \langle \mathbb{U}, \mu_{A\cup A}, \sigma_{A\cup A}, \omega_{A\cup A} \rangle \equiv \langle \mathbb{U}, \mu_A, \sigma_A, \omega_A \rangle \equiv \widetilde{A}$.

(2) $\forall u \in \mathbb{U}$ si ha che:

$$\begin{aligned} \mu_{A\cup\emptyset}(u) &= \mu_A(u) \vee \mu_\emptyset(u) \\ &= \mu_A(u) \vee \underline{0}(u) \\ &= \mu_A(u) \end{aligned}$$

$$\begin{aligned} \sigma_{A\cup\emptyset}(u) &= \sigma_A(u) \vee \sigma_\emptyset(u) \\ &= \sigma_A(u) \vee \underline{0}(u) \\ &= \sigma_A(u) \end{aligned}$$

$$\begin{aligned} \omega_{A\cup\emptyset}(u) &= \omega_A(u) \wedge \omega_\emptyset(u) \\ &= \omega_A(u) \wedge \underline{1}(u) \\ &= \omega_A(u) \end{aligned}$$

cosicché $\widetilde{A} \Cup \widetilde{\emptyset} \equiv \langle \mathbb{U}, \mu_{A\cup\emptyset}, \sigma_{A\cup\emptyset}, \omega_{A\cup\emptyset} \rangle \equiv \langle \mathbb{U}, \mu_A, \sigma_A, \omega_A \rangle \equiv \widetilde{A}$.

(3) Infatti $\forall u \in \mathbb{U}$ si ha che:

$$\begin{aligned} \mu_{A\cup\mathbb{U}}(u) &= \mu_A(u) \vee \mu_{\mathbb{U}}(u) \\ &= \mu_A(u) \vee 1 \\ &= 1 \\ &= \underline{1}(u) \end{aligned}$$

$$\begin{aligned} \sigma_{A\cup\mathbb{U}}(u) &= \sigma_A(u) \vee \sigma_{\mathbb{U}}(u) \\ &= \sigma_A(u) \vee 1 \\ &= 1 \\ &= \underline{1}(u) \end{aligned}$$

$$\begin{aligned} \omega_{A\cup\mathbb{U}}(u) &= \omega_A(u) \wedge \omega_{\mathbb{U}}(u) \\ &= \omega_A(u) \wedge 0 \\ &= 0 \\ &= \underline{0}(u) \end{aligned}$$

per cui $\widetilde{A} \Cup \widetilde{\mathbb{U}} \equiv \langle \mathbb{U}, \mu_{A\cup\mathbb{U}}, \sigma_{A\cup\mathbb{U}}, \omega_{A\cup\mathbb{U}} \rangle \equiv \langle \mathbb{U}, \underline{1}, \underline{1}, \underline{0} \rangle \equiv \widetilde{\mathbb{U}}$.

(4) Sia $u \in \mathbb{U}$ si ha che:

$$\begin{aligned} \mu_{A\cap A}(u) &= \mu_A(u) \wedge \mu_A(u) \\ &= \mu_A(u) \end{aligned}$$

$$\begin{aligned} \sigma_{A\cap A}(u) &= \sigma_A(u) \wedge \sigma_A(u) \\ &= \sigma_A(u) \end{aligned}$$

$$\begin{aligned} \omega_{A\cap A}(u) &= \omega_A(u) \vee \omega_A(u) \\ &= \omega_A(u) \end{aligned}$$

e quindi $\widetilde{A} \Cap \widetilde{A} \equiv \langle \mathbb{U}, \mu_{A\cap A}, \sigma_{A\cap A}, \omega_{A\cap A} \rangle \equiv \langle \mathbb{U}, \mu_A, \sigma_A, \omega_A \rangle \equiv \widetilde{A}$.

(5) Infatti $\forall u \in \mathbb{U}$ si ha che

$$\begin{aligned} \mu_{A\cap\emptyset}(u) &= \mu_A(u) \wedge \mu_{\emptyset}(u) \\ &= \mu_A(u) \wedge 0 \\ &= 0 \\ &= \underline{0}(u) \end{aligned}$$

$$\begin{aligned} \sigma_{A\cap\emptyset}(u) &= \sigma_A(u) \wedge \sigma_{\emptyset}(u) \\ &= \sigma_A(u) \wedge 0 \\ &= 0 \\ &= \underline{0}(u) \end{aligned}$$

$$\begin{aligned}\omega_{A\cap\emptyset}(u) &= \omega_A(u) \vee \omega_\emptyset(u)\\ &= \omega_A(u) \vee 1\\ &= 1\\ &= \underline{1}(u)\end{aligned}$$

e dunque che $\widetilde{A} \Cap \widetilde{\emptyset} \equiv \langle \mathbb{U}, \mu_{A\cap\emptyset}, \sigma_{A\cap\emptyset}, \omega_{A\cap\emptyset}\rangle \equiv \langle \mathbb{U}, \underline{0}, \underline{0}, \underline{1}\rangle \equiv \widetilde{\emptyset}$.

(6) Per ogni $u \in \mathbb{U}$ si ha che:

$$\begin{aligned}\mu_{A\cap\mathbb{U}}(u) &= \mu_A(u) \wedge \mu_{\mathbb{U}}(u)\\ &= \mu_A(u) \wedge 1\\ &= \mu_A(u)\end{aligned}$$

$$\begin{aligned}\sigma_{A\cap\mathbb{U}}(u) &= \sigma_A(u) \wedge \sigma_{\mathbb{U}}(u)\\ &= \sigma_A(u) \wedge 1\\ &= \sigma_A(u)\end{aligned}$$

$$\begin{aligned}\omega_{A\cap\mathbb{U}}(u) &= \omega_A(u) \vee \omega_{\mathbb{U}}(u)\\ &= \omega_A(u) \vee 0\\ &= \omega_A(u)\end{aligned}$$

e pertanto $\widetilde{A} \Cap \widetilde{\mathbb{U}} \equiv \langle \mathbb{U}, \mu_{A\cap\mathbb{U}}, \sigma_{A\cap\mathbb{U}}, \omega_{A\cap\mathbb{U}}\rangle \equiv \langle \mathbb{U}, \mu_A, \sigma_A, \omega_A\rangle \equiv \widetilde{A}$. □

Proposizione 2.2.9. [17] *Per ogni coppia $\widetilde{A} = \langle \mathbb{U}, \mu_A, \sigma_A, \omega_A\rangle$ e $\widetilde{B} = \langle \mathbb{U}, \mu_B, \sigma_B, \omega_B\rangle$ di SVN-set in $\mathcal{SVN}(\mathbb{U})$, si ha:*

(1) $\widetilde{A} \Cup \widetilde{B} \equiv \widetilde{B} \Cup \widetilde{A}$

(2) $\widetilde{A} \Cap \widetilde{B} \equiv \widetilde{B} \Cap \widetilde{A}$

Dimostrazione. (1) Sia $u \in \mathbb{U}$ si ha che:

$$\begin{aligned}\mu_{A\cup B}(u) &= \mu_A(u) \vee \mu_B(u)\\ &= \mu_B(u) \vee \mu_A(u)\\ &= \mu_{B\cup A}(u)\end{aligned}$$

$$\begin{aligned}\sigma_{A\cup B}(u) &= \sigma_A(u) \vee \sigma_B(u)\\ &= \sigma_B(u) \vee \sigma_A(u)\\ &= \sigma_{B\cup A}(u)\end{aligned}$$

$$\begin{aligned}\omega_{A\cup B}(u) &= \omega_A(u) \wedge \omega_B(u)\\ &= \omega_B(u) \wedge \omega_A(u)\\ &= \omega_{B\cup A}(u)\end{aligned}$$

cosicché $\widetilde{A} \Cup \widetilde{B} \equiv \langle \mathbb{U}, \mu_{A\cup B}, \sigma_{A\cup B}, \omega_{A\cup B}\rangle \equiv \langle \mathbb{U}, \mu_{B\cup A}, \sigma_{B\cup A}, \omega_{B\cup A}\rangle \equiv \widetilde{B} \Cup \widetilde{A}$.

(2) Per ogni $u \in \mathbb{U}$ si ha:

$$\begin{aligned}\mu_{A\cap B}(u) &= \mu_A(u) \wedge \mu_B(u)\\ &= \mu_B(u) \wedge \mu_A(u)\\ &= \mu_{B\cap A}(u)\end{aligned}$$

$$\begin{aligned}\sigma_{A\cap B}(u) &= \sigma_A(u) \wedge \sigma_B(u)\\ &= \sigma_B(u) \wedge \sigma_A(u)\\ &= \sigma_{B\cap A}(u)\end{aligned}$$

$$\begin{aligned}\omega_{A\cap B}(u) &= \omega_A(u) \vee \omega_B(u)\\ &= \omega_B(u) \vee \omega_A(u)\\ &= \omega_{B\cap A}(u)\end{aligned}$$

e pertanto $\widetilde{A} \Cap \widetilde{B} \equiv \langle \mathbb{U}, \mu_{A\cap B}, \sigma_{A\cap B}, \omega_{A\cap B}\rangle \equiv \langle \mathbb{U}, \mu_{B\cap A}, \sigma_{B\cap A}, \omega_{B\cap A}\rangle \equiv \widetilde{B} \Cap \widetilde{A}$. □

Proposizione 2.2.10. [17] *Per ogni terna* $\widetilde{A} = \langle \mathbb{U}, \mu_A, \sigma_A, \omega_A\rangle$, $\widetilde{B} = \langle \mathbb{U}, \mu_B, \sigma_B, \omega_B\rangle$ *e* $\widetilde{C} = \langle \mathbb{U}, \mu_C, \sigma_C, \omega_C\rangle$ *di SVN-set in* $\mathcal{SVN}(\mathbb{U})$*, si ha:*

(1) $\widetilde{A} \Cap \left(\widetilde{B} \Cap \widetilde{C}\right) \equiv \left(\widetilde{A} \Cap \widetilde{B}\right) \Cap \widetilde{C}$

(2) $\widetilde{A} \Cup \left(\widetilde{B} \Cup \widetilde{C}\right) \equiv \left(\widetilde{A} \Cup \widetilde{B}\right) \Cup \widetilde{C}$

Dimostrazione. (1) Per ogni $u \in \mathbb{U}$ allora risulta:

$$\begin{aligned}\mu_{A\cap(B\cap C)}(u) &= \mu_A(u) \wedge \mu_{B\cap C}(u)\\ &= \mu_A(u) \wedge (\mu_B(u) \wedge \mu_C(u))\\ &= (\mu_A(u) \wedge \mu_B(u)) \wedge \mu_C(u)\\ &= \mu_{A\cap B}(u) \wedge \mu_C(u)\\ &= \mu_{(A\cap B)\cap C}(u)\end{aligned}$$

$$\begin{aligned}\sigma_{A\cap(B\cap C)}(u) &= \sigma_A(u) \wedge \sigma_{B\cap C}(u)\\ &= \sigma_A(u) \wedge (\sigma_B(u) \wedge \sigma_C(u))\\ &= (\sigma_A(u) \wedge \sigma_B(u)) \wedge \sigma_C(u)\\ &= \sigma_{A\cap B}(u) \wedge \sigma_C(u)\\ &= \sigma_{(A\cap B)\cap C}(u)\end{aligned}$$

$$\begin{aligned}\omega_{A\cap(B\cap C)}(u) &= \omega_A(u) \vee \omega_{B\cap C}(u)\\ &= \omega_A(u) \vee (\omega_B(u) \vee \omega_C(u))\\ &= (\omega_A(u) \vee \omega_B(u)) \vee \omega_C(u)\\ &= \omega_{A\cap B}(u) \vee \omega_C(u)\\ &= \omega_{(A\cap B)\cap C}(u)\end{aligned}$$

e dunque:

$$\begin{aligned}\widetilde{A} \Cap \left(\widetilde{B} \Cap \widetilde{C}\right) &\equiv \langle \mathbb{U}, \mu_{A\cap(B\cap C)}, \sigma_{A\cap(B\cap C)}, \omega_{A\cap(B\cap C)}\rangle\\ &\equiv \langle \mathbb{U}, \mu_{(A\cap B)\cap C}, \sigma_{(A\cap B)\cap C}, \omega_{(A\cap B)\cap C}\rangle\\ &\equiv \left(\widetilde{A} \Cap \widetilde{B}\right) \Cap \widetilde{C}.\end{aligned}$$

(2) Per ogni $u \in \mathbb{U}$ allora possiamo considerare,

$$\begin{aligned}\mu_{A\cup(B\cup C)}(u) &= \mu_A(u) \vee (\mu_B(u) \vee \mu_C(u)) \\ &= (\mu_A(u) \vee \mu_B(u)) \vee \mu_C(u) \\ &= \mu_{(A\cup B)\cup C}(u)\end{aligned}$$

$$\begin{aligned}\sigma_{A\cup(B\cup C)}(u) &= \sigma_A(u) \vee (\sigma_B(u) \vee \sigma_C(u)) \\ &= (\sigma_A(u) \vee \sigma_B(u)) \vee \sigma_C(u) \\ &= \sigma_{(A\cup B)\cup C}(u)\end{aligned}$$

$$\begin{aligned}\omega_{A\cup(B\cup C)}(u) &= \omega_A(u) \wedge (\omega_B(u) \wedge \omega_C(u)) \\ &= (\omega_A(u) \wedge \omega_B(u)) \wedge \omega_C(u) \\ &= \omega_{(A\cup B)\cup C}(u)\end{aligned}$$

e quindi:

$$\begin{aligned}\widetilde{A} \Cup \left(\widetilde{B} \Cup \widetilde{C}\right) &\equiv \langle \mathbb{U}, \mu_{A\cup(B\cup C)}, \sigma_{A\cup(B\cup C)}, \omega_{A\cup(B\cup C)}\rangle \\ &\equiv \langle \mathbb{U}, \mu_{(A\cup B)\cup C}, \sigma_{(A\cup B)\cup C}, \omega_{(A\cup B)\cup C}\rangle \\ &\equiv \left(\widetilde{A} \Cup \widetilde{B}\right) \Cup \widetilde{C}.\end{aligned}$$

□

Proposizione 2.2.11. [17] *Siano* $\widetilde{A} = \langle \mathbb{U}, \mu_A, \sigma_A, \omega_A\rangle$, $\widetilde{B} = \langle \mathbb{U}, \mu_B, \sigma_B, \omega_B\rangle \in \mathcal{SVN}(\mathbb{U})$ *due SVN-set su un insieme universo* $\mathbb{U}$*, allora:*

(1) $\widetilde{A} \Subset \widetilde{B}$ *se e solo se* $\widetilde{A} \Cap \widetilde{B} \equiv \widetilde{A}$

(2) $\widetilde{A} \Subset \widetilde{B}$ *se e solo se* $\widetilde{A} \Cup \widetilde{B} \equiv \widetilde{B}$

Dimostrazione. (1) Supponiamo che $\widetilde{A} \Subset \widetilde{B}$ cioè che $\forall u \in \mathbb{U}$, $\mu_A(u) \leq \mu_B(u)$, $\sigma_A(u) \leq \sigma_B(u)$ e $\omega_A(u) \geq \omega_B(u)$. Allora $\forall u \in \mathbb{U}$, si ha:

$$\begin{aligned}\mu_{A\cap B}(u) &= \mu_A(u) \wedge \mu_B(u) \\ &= \mu_A(u)\end{aligned}$$

e analogamente

$$\begin{aligned}\sigma_{A\cap B}(u) &= \sigma_A(u) \wedge \sigma_B(u) \\ &= \sigma_A(u)\end{aligned}$$

$$\begin{aligned}\omega_{A\cap B}(u) &= \omega_A(u) \vee \mu_B(u) \\ &= \omega_A(u)\end{aligned}$$

cosicché $\widetilde{A} \Cap \widetilde{B} \equiv \langle \mathbb{U}, \mu_{A\cap B}, \sigma_{A\cap B}, \omega_{A\cap B}\rangle \equiv \langle \mathbb{U}, \mu_A, \sigma_A, \omega_A\rangle \equiv \widetilde{A}$.
Viceversa, se $\widetilde{A} \Cap \widetilde{B} \equiv \widetilde{A}$, $\forall u \in \mathbb{U}$ per la Definizione 2.2.4 di intersezione neutrosofica, si ha che:

$$\begin{aligned}\mu_{A\cap B}(u) &= \mu_A(u) \wedge \mu_B(u) \\ &= \mu_A(u)\end{aligned}$$

in particolare $\mu_A(u) \leq \mu_B(u)$, dal momento che $\mu_A(u) = \mu_A(u) \wedge \mu_B(u)$ significa che $\mu_A(u)$ è esattamente l'estremo inferiore tra $\mu_A(u)$ e $\mu_B(u)$. Analogamente

$$\begin{aligned}\sigma_{A\cap B}(u) &= \sigma_A(u) \wedge \sigma_B(u)\\ &= \sigma_A(u)\end{aligned}$$

e

$$\begin{aligned}\omega_{A\cap B}(u) &= \omega_A(u) \vee \mu_B(u)\\ &= \omega_A(u)\end{aligned}$$

dunque si ha che $\sigma_A(u) \leq \sigma_B(u)$ e $\omega_A(u) \geq \omega_B(u)$ e per ogni $u \in \mathbb{U}$ resta provata l'inclusione $\widetilde{A} \Subset \widetilde{B}$.

(2) Supponiamo che $\widetilde{A} \Subset \widetilde{B}$ cioè che $\forall u \in \mathbb{U}$, $\mu_A(u) \leq \mu_B(u)$, $\sigma_A(u) \leq \sigma_B(u)$ e $\omega_A(u) \geq \omega_B(u)$. Allora $\forall u \in \mathbb{U}$, si ha:

$$\begin{aligned}\mu_{A\cup B}(u) &= \mu_A(u) \vee \mu_B(u)\\ &= \mu_B(u)\end{aligned}$$

e analogamente

$$\begin{aligned}\sigma_{A\cup B}(u) &= \sigma_A(u) \vee \sigma_B(u)\\ &= \sigma_B(u)\end{aligned}$$

$$\begin{aligned}\omega_{A\cup B}(u) &= \omega_A(u) \wedge \mu_B(u)\\ &= \omega_B(u)\end{aligned}$$

cosicché $\widetilde{A} \Cup \widetilde{B} \equiv \langle \mathbb{U}, \mu_{A\cup B}, \sigma_{A\cup B}, \omega_{A\cup B}\rangle \equiv \langle \mathbb{U}, \mu_A, \sigma_A, \omega_A\rangle \equiv \widetilde{A}$.
Viceversa, se $\widetilde{A} \Cup \widetilde{B} \equiv \widetilde{B}$, $\forall u \in \mathbb{U}$, per la Definizione 2.2.3 di unione neutrosofica si ha che:

$$\begin{aligned}\mu_{A\cup B}(u) &= \mu_A(u) \vee \mu_B(u)\\ &= \mu_B(u)\end{aligned}$$

in particolare $\mu_A(u) \leq \mu_B(u)$, dal momento che $\mu_B(u) = \mu_A(u) \vee \mu_B(u)$ significa che $\mu_B(u)$ è esattamente l'estremo superiore tra $\mu_A(u)$ e $\mu_B(u)$. Analogamente

$$\begin{aligned}\sigma_{A\cup B}(u) &= \sigma_A(u) \vee \sigma_B(u)\\ &= \sigma_B(u)\end{aligned}$$

e

$$\begin{aligned}\omega_{A\cup B}(u) &= \omega_A(u) \wedge \mu_B(u)\\ &= \omega_B(u)\end{aligned}$$

dunque si ha che $\sigma_A(u) \leq \sigma_B(u)$ e $\omega_A(u) \geq \omega_B(u)$ per ogni $u \in \mathbb{U}$ e quindi resta provata l'inclusione $\widetilde{A} \Subset \widetilde{B}$. □

Proposizione 2.2.12. [17] *Per ogni coppia $\widetilde{A} = \langle \mathbb{U}, \mu_A, \sigma_A, \omega_A\rangle$ e $\widetilde{B} = \langle \mathbb{U}, \mu_B, \sigma_B, \omega_B\rangle$ di SVN-set in $\mathcal{SVN}(\mathbb{U})$, si ha:*

(1) $\widetilde{A} \Cup \left(\widetilde{A} \Cap \widetilde{B}\right) \equiv \widetilde{A}$

(2) $\widetilde{A} \Cap \left(\widetilde{A} \Cup \widetilde{B}\right) \equiv \widetilde{A}$

Dimostrazione. Poiché

$$\widetilde{A} \Cup \left(\widetilde{A} \Cap \widetilde{B}\right) = \left\langle \mathbb{U}, \mu_{A\cup(A\cap B)}, \sigma_{A\cup(A\cap B)}, \omega_{A\cup(A\cap B)} \right\rangle$$

e

$$\widetilde{A} \Cap \left(\widetilde{A} \Cup \widetilde{B}\right) = \left\langle \mathbb{U}, \mu_{A\cap(A\cup B)}, \sigma_{A\cap(A\cup B)}, \omega_{A\cap(A\cup B)} \right\rangle$$

per ogni $u \in \mathbb{U}$, bisogna provare che:

$$\begin{aligned} \mu_{A\cup(A\cap B)}(u) &= \mu_A(u) \\ \sigma_{A\cup(A\cap B)}(u) &= \sigma_A(u) \\ \omega_{A\cup(A\cap B)}(u) &= \omega_A(u) \end{aligned}$$

e:

$$\begin{aligned} \mu_{A\cap(A\cup B)}(u) &= \mu_A(u) \\ \sigma_{A\cap(A\cup B)}(u) &= \sigma_A(u) \\ \omega_{A\cap(A\cup B)}(u) &= \omega_A(u) \end{aligned}$$

(1) Infatti $\mu_{A\cup(A\cap B)}(u) = \mu_A(u)$, poiché se:

- $\mu_A(u) \leq \mu_B(u)$, allora $\mu_A(u) \vee (\mu_A(u) \wedge \mu_B(u)) = \mu_A(u) \vee \mu_A(u) = \mu_A(u)$.
- $\mu_A(u) \geq \mu_B(u)$, allora $\mu_A(u) \vee (\mu_A(u) \wedge \mu_B(u)) = \mu_A(u) \vee \mu_B(u) = \mu_A(u)$.

Analogamente, $\sigma_{A\cup(A\cap B)}(u) = \sigma_A(u)$, perché se:

- $\sigma_A(u) \leq \sigma_B(u)$, allora $\sigma_A(u) \vee (\sigma_A(u) \wedge \sigma_B(u)) = \sigma_A(u) \vee \sigma_A(u) = \sigma_A(u)$.
- $\sigma_A(u) \geq \sigma_B(u)$, allora $\sigma_A(u) \vee (\sigma_A(u) \wedge \sigma_B(u)) = \sigma_A(u) \vee \sigma_B(u) = \sigma_A(u)$.

e $\omega_{A\cup(A\cap B)}(u) = \omega_A(u)$, poiché se:

- $\omega_A(u) \leq \sigma_B(u)$, allora $\omega_A(u) \wedge (\omega_A(u) \vee \omega_B(u)) = \omega_A(u) \wedge \omega_B(u) = \omega_A(u)$.
- $\omega_A(u) \geq \omega_B(u)$, allora $\omega_A(u) \wedge (\omega_A(u) \vee \omega_B(u)) = \omega_A(u) \wedge \omega_A(u) = \omega_A(u)$.

Vale evidentemente anche il viceversa e quindi l'uguaglianza $\widetilde{A} \Cup \left(\widetilde{A} \Cap \widetilde{B}\right) \equiv \widetilde{A}$.

(2) Infatti $\mu_{A\cap(A\cup B)}(u) = \mu_A(u)$, poiché se:

- $\mu_A(u) \leq \mu_B(u)$, allora $\mu_A(u) \wedge (\mu_A(u) \vee \mu_B(u)) = \mu_A(u) \wedge \mu_B(u) = \mu_A(u)$.
- $\mu_A(u) \geq \mu_B(u)$, allora $\mu_A(u) \wedge (\mu_A(u) \vee \mu_B(u)) = \mu_A(u) \wedge \mu_A(u) = \mu_A(u)$.

Analogamente $\sigma_{A\cap(A\cup B)}(u) = \sigma_A(u)$, perché se:

- $\sigma_A(u) \leq \sigma_B(u)$, allora $\sigma_A(u) \wedge (\sigma_A(u) \vee \sigma_B(u)) = \sigma_A(u) \wedge \sigma_B(u) = \sigma_A(u)$.
- $\sigma_A(u) \geq \sigma_B(u)$, allora $\sigma_A(u) \wedge (\sigma_A(u) \vee \sigma_B(u)) = \sigma_A(u) \wedge \sigma_A(u) = \sigma_A(u)$.

e $\omega_{A\cap(A\cup B)}(u) = \omega_A(u)$, poiché se:

- $\omega_A(u) \leq \sigma_B(u)$, allora $\omega_A(u) \vee (\omega_A(u) \wedge \omega_B(u)) = \omega_A(u) \vee \omega_A(u) = \omega_A(u)$.
- $\omega_A(u) \geq \omega_B(u)$, allora $\omega_A(u) \vee (\omega_A(u) \wedge \omega_B(u)) = \omega_A(u) \vee \omega_B(u) = \omega_A(u)$.

Vale evidentemente anche il viceversa e quindi l'uguaglianza $\widetilde{A} \Cap \left(\widetilde{A} \Cup \widetilde{B}\right) \equiv \widetilde{A}$. $\square$

Proposizione 2.2.13. [17] *Siano* $\widetilde{A} = \langle \mathbb{U}, \mu_A, \sigma_A, \omega_A \rangle$, $\widetilde{B} = \langle \mathbb{U}, \mu_B, \sigma_B, \omega_B \rangle$, $\widetilde{C} = \langle \mathbb{U}, \mu_C, \sigma_C, \omega_C \rangle$ *e* $\widetilde{D} = \langle \mathbb{U}, \mu_D, \sigma_D, \omega_D \rangle$ *SVN-set in* $\mathcal{SVN}(\mathbb{U})$ *tali che* $\widetilde{A} \Subset \widetilde{B}$ *e* $\widetilde{C} \Subset \widetilde{D}$, *allora:*

(1) $\widetilde{A} \Cup \widetilde{C} \Subset \widetilde{B} \Cup \widetilde{D}$

(2) $\widetilde{A} \Cap \widetilde{C} \Subset \widetilde{B} \Cap \widetilde{D}$

Dimostrazione. Siano $\widetilde{A} \Subset \widetilde{B}$ e $\widetilde{C} \Subset \widetilde{D}$. Allora per ogni $u \in \mathbb{U}$ si ha che:

$$\mu_A(u) \leq \mu_B(u), \quad \sigma_A(u) \leq \sigma_B(u) \quad \text{e} \quad \omega_A(u) \geq \omega_B(u)$$

e

$$\mu_C(u) \leq \mu_D(u), \quad \sigma_C(u) \leq \sigma_D(u) \quad \text{e} \quad \omega_C(u) \geq \omega_D(u)$$

e dunque risulta:
(1)

$$\begin{aligned} \mu_{A \cup B}(u) &= \mu_A(u) \vee \mu_C(u) \\ &\leq \mu_B(u) \vee \mu_D(u) \\ &= \mu_{B \cup D}(u) \end{aligned}$$

e analogamente

$$\begin{aligned} \sigma_{A \cup B}(u) &= \sigma_A(u) \vee \sigma_C(u) \\ &\leq \sigma_B(u) \vee \sigma_D(u) \\ &= \sigma_{B \cup D}(u) \end{aligned}$$

$$\begin{aligned} \omega_{A \cup B}(u) &= \omega_A(u) \wedge \omega_C(u) \\ &\geq \omega_B(u) \wedge \omega_D(u) \\ &= \omega_{B \cup D}(u) \end{aligned}$$

e pertanto per la Definizione 2.1.2 di sottoinsieme neutrosofico, ne segue che $\widetilde{A} \Cup \widetilde{C} \Subset \widetilde{B} \Cup \widetilde{D}$.

(2)

$$\begin{aligned} \mu_{A \cap B}(u) &= \mu_A(u) \wedge \mu_C(u) \\ &\leq \mu_B(u) \wedge \mu_D(u) \\ &= \mu_{B \cap D}(u) \end{aligned}$$

e analogamente

$$\begin{aligned} \sigma_{A \cap B}(u) &= \sigma_A(u) \wedge \sigma_C(u) \\ &\leq \sigma_B(u) \wedge \sigma_D(u) \\ &= \sigma_{B \cap D}(u) \end{aligned}$$

$$\begin{aligned}\omega_{A\cap B}(u) &= \omega_A(u) \vee \omega_C(u)\\ &\geq \omega_B(u) \vee \omega_D(u)\\ &= \omega_{B\cap D}(u)\end{aligned}$$

e pertanto per la Definizione 2.1.2 di sottoinsieme neutrosofico, ne segue che $\widetilde{A} \Subset \widetilde{C} \subseteqq \widetilde{B} \Subset \widetilde{D}$.

□

Proposizione 2.2.14. [17] *Sia* $\left\{\widetilde{A}_i\right\}_{i\in I}$ *una famiglia di SVN-set* $\widetilde{A}_i = \langle \mathbb{U}, \mu_{A_i}, \sigma_{A_i}, \omega_{A_i}\rangle$ *su uno stesso insieme universo* $\mathbb{U}$*, allora, per ogni* $i \in I$*, si ha che* $\bigcap_{i\in I} \widetilde{A}_i \subseteqq \widetilde{A}_i \subseteqq \bigcup_{i\in I} \widetilde{A}_i$.

Dimostrazione. Infatti, per ogni $u \in \mathbb{U}$ è evidente che:

$$\begin{aligned}\inf_{i\in I} \mu_{A_i}(u) \leq \mu_{A_i}(u) \leq \sup_{i\in I} \mu_{A_i}(u)\\ \inf_{i\in I} \sigma_{A_i}(u) \leq \sigma_{A_i}(u) \leq \sup_{i\in I} \sigma_{A_i}(u)\\ \sup_{i\in I} \omega_{A_i}(u) \geq \omega_{A_i}(u) \geq \inf_{i\in I} \omega_{A_i}(u)\end{aligned}$$

e dunque per la Definizione 2.1.2 di sottoinsieme neutrosofico, si ha che $\bigcap_{i\in I} \widetilde{A}_i \subseteqq \widetilde{A}_i \subseteqq \bigcup_{i\in I} \widetilde{A}_i$. □

Proposizione 2.2.15. [17] *Siano rispettivamente* $\widetilde{A} = \langle \mathbb{U}, \mu_A, \sigma_A, \omega_A\rangle \in \mathcal{SVN}(\mathbb{U})$ *un SVN-set e* $\left\{\widetilde{B}_i\right\}_{i\in I} \subseteq \mathcal{SVN}(\mathbb{U})$ *una famiglia di SVN-set* $\widetilde{B}_i = \langle \mathbb{U}, \mu_{B_i}, \sigma_{B_i}, \omega_{B_i}\rangle$ *su uno stesso insieme universo* $\mathbb{U}$*, allora si ha:*

(1) $\widetilde{A} \Subset \left(\bigcup_{i\in I} \widetilde{B}_i\right) \equiv \bigcup_{i\in I} \left(\widetilde{A} \Subset \widetilde{B}_i\right)$

(2) $\widetilde{A} \Cup \left(\bigcap_{i\in I} \widetilde{B}_i\right) \equiv \bigcap_{i\in I} \left(\widetilde{A} \Cup \widetilde{B}_i\right)$

Dimostrazione. (1) Per ogni $u \in \mathbb{U}$ si ha che:

$$\begin{aligned}\mu_{A\cap(\bigcup_{i\in I} B_i)}(u) &= \mu_A(u) \wedge (\sup_{i\in I} \mu_{B_i}(u))\\ &= \sup_{i\in I}(\mu_A(u) \wedge \mu_{B_i}(u))\\ &= \mu_{\bigcup_{i\in I}(A\cap B_i)}(u).\end{aligned}$$

e analogamente:

$$\begin{aligned}\sigma_{A\cap(\bigcup_{i\in I} B_i)}(u) &= \sigma_A(u) \wedge (\sup_{i\in I} \sigma_{B_i}(u))\\ &= \sup_{i\in I}(\sigma_A(u) \wedge \sigma_{B_i}(u))\\ &= \sigma_{\bigcup_{i\in I}(A\cap B_i)}(u).\end{aligned}$$

$$\begin{aligned}\omega_{A\cap(\bigcup_{i\in I} B_i)}(u) &= \omega_A(u) \vee (\inf_{i\in I} \omega_{B_i}(u))\\ &= \inf_{i\in I}(\omega_A(u) \vee \omega_{B_i}(u))\\ &= \omega_{\bigcup_{i\in I}(A\cap B_i)}(u).\end{aligned}$$

e dunque $\widetilde{A} \Cap \left(\bigcup_{i\in I} \widetilde{B}_i\right) \equiv \bigcup_{i\in I} \left(\widetilde{A} \Cap \widetilde{B}_i\right)$.

(2) Per ogni $u \in \mathbb{U}$, si ha che:

$$\begin{aligned}\mu_{A\cup(\bigcap_{i\in I}\mu_{B_i}(u))}(u) &= \mu_A(u) \vee (\inf_{i\in I}\mu_{B_i}(u)) \\ &= \inf_{i\in I}(\mu_A(u) \vee \mu_{B_i}(u)) \\ &= \mu_{\bigcap_{i\in I}(A\cup B_i)}(u).\end{aligned}$$

e analogamente

$$\begin{aligned}\sigma_{A\cup(\bigcap_{i\in I}\sigma_{B_i}(u))}(u) &= \sigma_A(u) \vee (\inf_{i\in I}\sigma_{B_i}(u)) \\ &= \inf_{i\in I}(\sigma_A(u) \vee \sigma_{B_i}(u)) \\ &= \sigma_{\bigcap_{i\in I}(A\cup B_i)}(u).\end{aligned}$$

$$\begin{aligned}\omega_{A\cup(\bigcap_{i\in I}\omega_{B_i}(u))}(u) &= \omega_A(u) \wedge (\sup_{i\in I}\omega_{B_i}(u)) \\ &= \sup_{i\in I}(\omega_A(u) \wedge \omega_{B_i}(u)) \\ &= \omega_{\bigcap_{i\in I}(A\cup B_i)}(u).\end{aligned}$$

e quindi $\widetilde{A} \Cup \left(\bigcap_{i\in I} \widetilde{B}_i\right) \equiv \bigcap_{i\in I} \left(\widetilde{A} \Cup \widetilde{B}_i\right)$. □

Proposizione 2.2.16. [17] *Sia* $\left\{\widetilde{A}_i\right\}_{i\in I} \subseteq \mathcal{SVN}(\mathbb{U})$ *una famiglia di SVN-set* $\widetilde{A}_i = \langle \mathbb{U}, \mu_{A_i}, \sigma_{A_i}, \omega_{A_i}\rangle$ *su uno stesso insieme universo* $\mathbb{U}$*, risulta che:*

(1) $\left(\bigcup_{i\in I} \widetilde{A}_i\right)^{\complement} \equiv \bigcap_{i\in I} \widetilde{A}_i^{\complement}$

(2) $\left(\bigcap_{i\in I} \widetilde{A}_i\right)^{\complement} \equiv \bigcup_{i\in I} \widetilde{A}_i^{\complement}$

Dimostrazione. (1) Sia $\widetilde{A_i} = \langle \mathbb{U}, \mu_{A_i}, \sigma_{A_i}, \omega_{A_i}\rangle$ per ogni $i \in I$, allora detto $\widetilde{A} = \bigcup_{i\in I} \widetilde{A_i} = \langle \mathbb{U}, \mu_A, \sigma_A, \omega_A\rangle$ con

$$\mu_A = \sup_{i\in I} \mu_{A_i}, \quad \sigma_A = \sup_{i\in I} \sigma_{A_i} \quad \text{e} \quad \omega_A = \inf_{i\in I} \omega_{A_i}$$

allora posto $\widetilde{C} = \widetilde{A}^{\complement} = \langle \mathbb{U}, \mu_C, \sigma_C, \omega_C\rangle$ ne segue che:

$$\begin{aligned}\mu_C &= 1 - \mu_A \\ &= 1 - \sup_{i\in I} \mu_{A_i} \\ &= \inf_{i\in I} (1 - \mu_{A_i})\end{aligned}$$

$$\begin{aligned}\sigma_C &= 1 - \sigma_A \\ &= 1 - \sup_{i\in I} \sigma_{A_i} \\ &= \inf_{i\in I} (1 - \sigma_{A_i})\end{aligned}$$

$$\begin{aligned}\omega_C &= 1 - \omega_A \\ &= 1 - \inf_{i\in I} \omega_{A_i} \\ &= \sup_{i\in I} \omega_{A_i}\end{aligned}$$

Dalla Definizione 2.2.1 sappiamo che $\widetilde{A_i}^{\complement} = \langle \mathbb{U}, \underline{1} - \omega_{A_i}, \underline{1} - \sigma_{A_i}, \underline{1} - \mu_{A_i} \rangle$ per ogni $i \in I$. Detto $\widetilde{B} = \bigcap_{i\in I} \widetilde{A_i}^{\complement} = \langle \mathbb{U}, \mu_B, \sigma_B, \omega_B \rangle$ con

$$\mu_B = \inf_{i\in I} (1 - \mu_{A_i}), \quad \sigma_B = \inf_{i\in I} (1 - \sigma_{A_i}) \quad \text{e} \quad \omega_B = \sup_{i\in I} (1 - \omega_{A_i})$$

risulta $\mu_C = \mu_B$, $\sigma_C = \sigma_B$ e $\omega_C = \omega_B$, cosicché resta provata l'uguaglianza $\left(\bigcup_{i\in I} \widetilde{A}_i\right)^{\complement} \equiv \bigcap_{i\in I} \widetilde{A}_i^{\complement}$

(2) Sia $\widetilde{A_i} = \langle \mathbb{U}, \mu_{A_i}, \sigma_{A_i}, \omega_{A_i} \rangle$ per ogni $i \in I$, dunque detto $\widetilde{A} = \bigcap_{i\in I} \widetilde{A_i} = \langle \mathbb{U}, \mu_A, \sigma_A, \omega_A \rangle$ con

$$\mu_A = \inf_{i\in I} \mu_{A_i}, \quad \sigma_A = \inf_{i\in I} \sigma_{A_i} \quad \text{e} \quad \omega_A = \sup_{i\in I} \omega_{A_i}$$

allora posto $\widetilde{C} = \widetilde{A}^{\complement} = \langle \mathbb{U}, \mu_C, \sigma_C, \omega_C \rangle$ si ha che:

$$\begin{aligned}\mu_C &= 1 - \mu_A \\ &= 1 - \inf_{i\in I} \mu_{A_i} \\ &= \sup_{i\in I} (1 - \mu_{A_i})\end{aligned}$$

$$\begin{aligned}\sigma_C &= 1 - \sigma_A \\ &= 1 - \inf_{i\in I} \sigma_{A_i} \\ &= \sup_{i\in I} (1 - \sigma_{A_i})\end{aligned}$$

$$\begin{aligned}\omega_C &= 1 - \omega_A \\ &= 1 - \sup_{i\in I} \omega_{A_i} \\ &= \inf_{i\in I} (1 - \omega_{A_i})\end{aligned}$$

Dalla Definizione 2.2.1 sappiamo che $\widetilde{A_i}^{\complement} = \langle \mathbb{U}, \underline{1} - \mu_{A_i}, \underline{1} - \sigma_{A_i}, \underline{1} - \omega_{A_i} \rangle$ per ogni $i \in I$. Detto $\widetilde{B} = \bigcup_{i\in I} \widetilde{A_i}^{\complement} = \langle \mathbb{U}, \mu_B, \sigma_B, \omega_B \rangle$ con

$$\mu_B = \sup_{i\in I} (1 - \mu_{A_i}), \quad \sigma_B = \sup_{i\in I} (1 - \sigma_{A_i}) \quad \text{e} \quad \omega_B = \inf_{i\in I} (1 - \omega_{A_i})$$

si ha che $\mu_C = \mu_B$, $\sigma_C = \sigma_B$ e $\omega_C = \omega_B$, dunque resta provata l'uguaglianza $\left(\bigcap_{i\in I} \widetilde{A}_i\right)^{\complement} \equiv \bigcup_{i\in I} \widetilde{A}_i^{\complement}$ □

2.3 Funzioni neutrosofiche

Definizione 2.3.1. [7, 10] *Sia $f : \mathbb{U} \to \mathbb{V}$ una funzione tra due insiemi universo $\mathbb{U}$ e $\mathbb{V}$, e sia $\widetilde{A} = \langle \mathbb{U}, \mu_A, \sigma_A, \omega_A \rangle$ un SVN-set su $\mathbb{U}$. L'* ***immagine neutrosofica*** *di $\widetilde{A}$ tramite f, indicata con $\widetilde{f}\left(\widetilde{A}\right)$, è il SVN-set su $\mathbb{V}$ definito da:*

$$\widetilde{f}\left(\widetilde{A}\right) = \left\langle \mathbb{V}, \mu_{f(A)}, \sigma_{f(A)}, \omega_{f(A)} \right\rangle$$

dove le funzioni $\mu_{f(A)} : \mathbb{V} \to I$, $\sigma_{f(A)} : \mathbb{V} \to I$ and $\omega_{f(A)} : \mathbb{V} \to I$ sono definite rispettivamente da:

$$\mu_{f(A)}(v) = \begin{cases} \sup\limits_{u \in f^{-1}(\{v\})} \mu_A(u) & \text{se } f^{-1}(\{v\}) \neq \emptyset \\ 1 & \text{altrimenti} \end{cases},$$

$$\sigma_{f(A)}(v) = \begin{cases} \sup\limits_{u \in f^{-1}(\{v\})} \sigma_A(u) & \text{se } f^{-1}(\{v\}) \neq \emptyset \\ 1 & \text{altrimenti} \end{cases},$$

$$\omega_{f(A)}(v) = \begin{cases} \inf\limits_{u \in f^{-1}(\{v\})} \omega_A(u) & \text{se } f^{-1}(\{v\}) \neq \emptyset \\ 0 & \text{altrimenti} \end{cases}$$

per ogni $v \in \mathbb{V}$.

Definizione 2.3.2. [7, 10] *Sia $f : \mathbb{U} \to \mathbb{V}$ una funzione tra due insiemi universo $\mathbb{U}$ e $\mathbb{V}$, e sia $\widetilde{B} = \langle \mathbb{V}, \mu_B, \sigma_B, \omega_B \rangle$ un SVN-set su $\mathbb{V}$. La* ***retroimmagine neutrosofica*** *di $\widetilde{B}$ tramite f, indicata con $\widetilde{f}^{-1}\left(\widetilde{B}\right)$, è il SVN-set su $\mathbb{U}$ definito da:*

$$\widetilde{f}^{-1}\left(\widetilde{B}\right) = \left\langle \mathbb{U}, \mu_{f^{-1}(B)}, \sigma_{f^{-1}(B)}, \omega_{f^{-1}(B)} \right\rangle$$

dove le funzioni $\mu_{f^{-1}(B)} : \mathbb{U} \to I$, $\sigma_{f^{-1}(B)} : \mathbb{U} \to I$ e $\omega_{f^{-1}(B)} : \mathbb{U} \to I$ sono definite rispettivamente da:

$$\mu_{f^{-1}(B)}(u) = \mu_B(f(u)),$$

$$\sigma_{f^{-1}(B)}(u) = \sigma_B(f(u)),$$

$$\omega_{f^{-1}(B)}(u) = \omega_B(f(u))$$

per ogni $u \in \mathbb{U}$.

Osservazione 2.3.3. *Si osservi che la notazione usata per denotare le immagini neutrosofiche e le retroimmagini neutrosofiche implicitamente sottolinea il fatto che si considera una funzione $\widetilde{f} : \mathcal{SVN}(\mathbb{U}) \to \mathcal{SVN}(\mathbb{V})$ tra tutti i possibili SVN-set su $\mathbb{U}$ e su $\mathbb{V}$, indotta dalla funzione $f : \mathbb{U} \to \mathbb{V}$ che invece è definita da $\mathbb{U}$ in $\mathbb{V}$. Più precisamente, la $\widetilde{f}$ è una funzione tra tutti i possibili SVN-set su $\mathbb{U}$ e su $\mathbb{V}$, mentre la f è una funzione fra gli elementi di $\mathbb{U}$ e $\mathbb{V}$.*

Esempio 2.3.4. [8] *Siano $\mathbb{U} = \{a, b, c\}$ e $\mathbb{V} = \{\alpha, \beta, \gamma, \delta\}$ due insiemi universo finiti, $f : \mathbb{U} \to \mathbb{V}$ una funzione definita da $f(a) = f(c) = \beta$ e $f(b) = \alpha$. Consideriamo un SVN-set $\widetilde{A} = \langle \mathbb{U}, \mu_A, \sigma_A, \omega_A \rangle$ su $\mathcal{SVN}(\mathbb{U})$ ed un SVN-set $\widetilde{B} = \langle \mathbb{U}, \mu_B, \sigma_B, \omega_B \rangle$ su $\mathcal{SVN}(\mathbb{V})$ rispettivamente definiti dalle seguenti rappresentazioni tabulari:*

$\mathbb{U}$ \ $\widetilde{A}$	μ_A	σ_A	ω_A
a	*0.5*	*0.2*	*0.2*
b	*0.6*	*0.2*	*0.3*
c	*0.4*	*0.3*	*0.1*

$\mathbb{V}$ \ $\widetilde{B}$	μ_B	σ_B	ω_B
α	*0.1*	*0.7*	*0.9*
β	*0.5*	*0.3*	*0.1*
γ	*0.8*	*0.4*	*0.2*
δ	*0.4*	*0.6*	*0.8*

Allora si ha che l'immagine neutrosofica $\widetilde{f}\left(\widetilde{A}\right) = \left\langle \mathbb{V}, \mu_{f(A)}, \sigma_{f(A)}, \omega_{f(A)} \right\rangle$ *di* $\widetilde{A}$ *tramite* f *e la retroimmagine neutrosofica* $\widetilde{f}^{-1}\left(\widetilde{B}\right) = \left\langle \mathbb{U}, \mu_{f^{-1}(B)}, \sigma_{f^{-1}(B)}, \omega_{f^{-1}(B)} \right\rangle$ *di* $\widetilde{B}$ *tramite* f *sono date rispettivamente da:*

$\mathbb{V}$ \ $\widetilde{f}\left(\widetilde{A}\right)$	$\mu_{f(A)}$	$\sigma_{f(A)}$	$\omega_{f(A)}$
α	*0.6*	*0.2*	*0.3*
β	*0.5*	*0.3*	*0.1*
γ	*1*	*1*	*0*
δ	*1*	*1*	*0*

e

$\mathbb{U}$ \ $\widetilde{f}^{-1}\left(\widetilde{B}\right)$	$\mu_{f^{-1}(B)}$	$\sigma_{f^{-1}(B)}$	$\omega_{f^{-1}(B)}$
a	*0.5*	*0.3*	*0.1*
b	*0.1*	*0.7*	*0.9*
c	*0.5*	*0.3*	*0.1*

Proposizione 2.3.5. [10] *Siano* $f : \mathbb{U} \to \mathbb{V}$ *una funzione tra due insiemi universo* $\mathbb{U}$ *e* $\mathbb{V}$*,* $\widetilde{A} = \langle \mathbb{U}, \mu_A, \sigma_A, \omega_A \rangle$ *un SVN-set su* $\mathbb{U}$ *e* $\widetilde{B} = \langle \mathbb{V}, \mu_B, \sigma_B, \omega_B \rangle$ *un SVN-set su* $\mathbb{V}$*, allora valgono le seguenti:*

(1) $\widetilde{f}\left(\widetilde{\emptyset}_{\mathbb{U}}\right) \equiv \widetilde{\emptyset}_{\mathbb{V}}$

(2) $\widetilde{f}^{-1}\left(\widetilde{\emptyset}_{\mathbb{V}}\right) \equiv \widetilde{\emptyset}_{\mathbb{U}}$

(3) $\widetilde{f}^{-1}\left(\widetilde{\mathbb{V}}\right) \equiv \widetilde{\mathbb{U}}$

(4) $\widetilde{A} \Subseteq \widetilde{f}^{-1}\left(\widetilde{f}\left(\widetilde{A}\right)\right)$ *e vale l'uguaglianza se* $\widetilde{f}$ *è iniettiva*

(5) $\widetilde{f}\left(\widetilde{f}^{-1}\left(\widetilde{B}\right)\right) \Subseteq \widetilde{B}$ *e vale l'uguaglianza se* $\widetilde{f}$ *se è suriettiva*

(6) $\widetilde{f}^{-1}\left(\widetilde{B}^{\complement}\right) \equiv \widetilde{f}^{-1}\left(\widetilde{B}\right)^{\complement}$

Dimostrazione. (1) Dalla Definizione 2.3.1 si ha che $\forall v \in \mathbb{V}$ se $f^{-1}(\{v\}) \neq \emptyset$ risultano:

$$\begin{aligned}\mu_{f(\emptyset_{\mathbb{U}})}(v) &= \sup_{u \in f^{-1}(\{v\})} \mu_{\emptyset_{\mathbb{U}}}(u) \\ &= \sup_{u \in f^{-1}(\{v\})} \underline{1}(u) \\ &= 1 \\ &= \underline{1}(v)\end{aligned}$$

$$\begin{aligned}\sigma_{f(\emptyset_{\mathbb{U}})}(v) &= \sup_{u \in f^{-1}(\{v\})} \sigma_{\emptyset_{\mathbb{U}}}(u) \\ &= \sup_{u \in f^{-1}(\{v\})} \underline{1}(u) \\ &= 1 \\ &= \underline{1}(v)\end{aligned}$$

$$\begin{aligned}\omega_{f(\emptyset_{\mathbb{U}})}(v) &= \inf_{u \in f^{-1}(\{v\})} \omega_{\emptyset_{\mathbb{U}}}(u) \\ &= \inf_{u \in f^{-1}(\{v\})} \underline{1}(u) \\ &= 0 \\ &= \underline{0}(v)\end{aligned}$$

inoltre se $f^{-1}(\{v\}) = \emptyset$ risulta altresì

$$\mu_{f(\emptyset_{\mathbb{U}})}(v) = 1, \quad \sigma_{f(\emptyset_{\mathbb{U}})}(v), \quad \omega_{f(\emptyset_{\mathbb{U}})}(v) = 0$$

e quindi che $\widetilde{f}\left(\widetilde{\emptyset}_{\mathbb{U}}\right) \equiv \langle \mathbb{V}, \underline{1}, \underline{1}, \underline{0} \rangle \equiv \widetilde{\emptyset}_{\mathbb{V}}$.

(2) Detto $\widetilde{f}^{-1}\left(\widetilde{\emptyset}_{\mathbb{V}}\right) = \left\langle \mathbb{U}, \mu_{f^{-1}(\emptyset_{\mathbb{V}})}, \sigma_{f^{-1}(\emptyset_{\mathbb{V}})}, \omega_{f^{-1}(\emptyset_{\mathbb{V}})} \right\rangle$, $\forall u \in \mathbb{U}$ si ha che:

$$\begin{aligned}\mu_{f^{-1}(\widetilde{\emptyset}_{\mathbb{V}})}(u) &= \mu_{\widetilde{\emptyset}_{\mathbb{V}}}(f(u)) \\ &= \underline{0}(f(u)) \\ &= 0 \\ &= \underline{0}(u)\end{aligned}$$

$$\begin{aligned}\sigma_{f^{-1}(\widetilde{\emptyset}_{\mathbb{V}})}(u) &= \sigma_{\widetilde{\emptyset}_{\mathbb{V}}}(f(u)) \\ &= \underline{0}(f(u)) \\ &= 0 \\ &= \underline{0}(u)\end{aligned}$$

$$\begin{aligned}\omega_{f^{-1}(\widetilde{\emptyset}_{\mathbb{V}})}(u) &= \omega_{\widetilde{\emptyset}_{\mathbb{V}}}(f(u)) \\ &= \underline{1}(f(u)) \\ &= 1 \\ &= \underline{1}(u)\end{aligned}$$

e quindi risulta $\widetilde{f}^{-1}\left(\widetilde{\emptyset}_{\mathbb{V}}\right) \equiv \langle \mathbb{U}, \underline{0}, \underline{0}, \underline{1} \rangle \equiv \widetilde{\emptyset}_{\mathbb{U}}$.

(3) Detto $\widetilde{f}^{-1}(\mathbb{V}) = \left\langle \mathbb{U}, \mu_{f^{-1}(\mathbb{V})}, \sigma_{f^{-1}(\mathbb{V})}, \omega_{f^{-1}(\mathbb{V})} \right\rangle$, $\forall u \in \mathbb{U}$ si ha che:

$$\begin{aligned} \mu_{f^{-1}(\mathbb{V})}(u) &= \mu_{\mathbb{V}}\left(f(u)\right) \\ &= \underline{1}(f(u)) \\ &= 1 \\ &= \underline{1}(u) \end{aligned}$$

$$\begin{aligned} \sigma_{f^{-1}(\mathbb{V})}(u) &= \sigma_{\mathbb{V}}\left(f(u)\right) \\ &= \underline{1}(u) \\ &= 1 \\ &= \underline{1}(u) \end{aligned}$$

$$\begin{aligned} \omega_{f^{-1}(\mathbb{V})}(u) &= \omega_{\mathbb{V}}\left(f(u)\right) \\ &= \underline{0}(f(u)) \\ &= 0 \\ &= \underline{0}(u) \end{aligned}$$

e dunque $\widetilde{f}^{-1}(\mathbb{V}) \equiv \mathbb{U}$

(4) Proviamo che $\widetilde{A} \Subset \widetilde{f}^{-1}\left(\widetilde{f}\left(\widetilde{A}\right)\right)$, verificando che $\forall u \in \mathbb{U}$ risulta:

$$\begin{aligned} \mu_A(u) &\leq \mu_{f^{-1}(f(A))}(u) \\ \sigma_A(u) &\leq \sigma_{f^{-1}(f(A))}(u) \\ \omega_A(u) &\geq \omega_{f^{-1}(f(A))}(u) \end{aligned}$$

Osserviamo che $\widetilde{f}^{-1}\left(\widetilde{f}\left(\widetilde{A}\right)\right) = \left\langle \mathbb{U}, \mu_{f^{-1}(f(A))}, \sigma_{f^{-1}(f(A))}, \omega_{f^{-1}(f(A))} \right\rangle$ con

$$\begin{aligned} \mu_{f^{-1}(f(A))} &= \mu_{f(A)} \circ f \\ \sigma_{f^{-1}(f(A))} &= \sigma_{f(A)} \circ f \\ \omega_{f^{-1}(f(A))} &= \omega_{f(A)} \circ f \end{aligned}$$

cioè che per ogni $u \in \mathbb{U}$

$$\begin{aligned} \mu_{f^{-1}(f(A))}(u) &= \mu_{f(A)} \circ f(u) \\ \sigma_{f^{-1}(f(A))}(u) &= \sigma_{f(A)} \circ f(u) \\ \omega_{f^{-1}(f(A))}(u) &= \omega_{f(A)} \circ f(u) \end{aligned}$$

Per la Definizione 2.3.1, $\forall t \in f^{-1}\{f(u)\} \neq \emptyset$ si ha che

$$\begin{aligned} \mu_A(t) &\leq \mu_{f(A)}(f(t)) \\ \sigma_A(t) &\leq \sigma_{f(A)}(f(t)) \\ \omega_A(t) &\geq \omega_{f(A)}(f(t)) \end{aligned}$$

e poiché in particolare $u \in f^{-1}\{f(u)\}$ ne segue che

$$\begin{aligned}\mu_A(u) &\leq \mu_{f(A)}(f(u)) \\ &= \mu_{f^{-1}(f(A))}(u)\end{aligned}$$

$$\begin{aligned}\sigma_A(u) &\leq \sigma_{f(A)}(f(u)) \\ &= \sigma_{f^{-1}(f(A))}(u)\end{aligned}$$

$$\begin{aligned}\omega_A(u) &\geq \omega_{f(A)}(f(u)) \\ &= \omega_{f^{-1}(f(A))}(u)\end{aligned}$$

Inoltre, se la $\widetilde{f}$ è iniettiva risulta $f^{-1}(f(u)) = u$, $\forall u \in \mathbb{U}$. Quindi da questo fatto discende che $\widetilde{A} = \widetilde{f}^{-1}\left(\widetilde{f}\left(\widetilde{A}\right)\right)$, cioè la tesi.

(5) Detto $\widetilde{B} = \langle \mathbb{V}, \mu_B, \sigma_B, \omega_B \rangle$ e posto $\widetilde{f}\left(\widetilde{f}^{-1}\left(\widetilde{B}\right)\right) = \left\langle \mathbb{V}, \mu_{f(f^{-1}(B))}, \sigma_{f(f^{-1}(B))}, \omega_{f(f^{-1}(B))} \right\rangle$, dobbiamo verificare che $\forall v \in \mathbb{V}$ risulta:

$$\begin{aligned}&\mu_{f(f^{-1}(B))}(v) \leq \mu_B(v), \\ &\sigma_{f(f^{-1}(B))}(v) \leq \sigma_B(v)\end{aligned}$$

e

$$\mu_{f(f^{-1}(B))}(v) \leq \mu_B(v)$$

Ora, per la Definizione 2.3.1, $\forall v \in \mathbb{V}$ si ha che:

$$\mu_{f(f^{-1}(B))}(v) = \begin{cases} \sup\limits_{u \in f^{-1}(\{v\})} \mu_{f^{-1}(B)}(u) & \text{se } f^{-1}(\{v\}) \neq \emptyset \\ 1 & \text{altrimenti} \end{cases},$$

e poiché per la Definizione 2.3.2 si ha che $\mu_{f^{-1}(B)}(u) = \mu_B(f(u))$, ne segue che se $f^{-1}(\{v\}) \neq \emptyset$ risulta:

$$\begin{aligned}\mu_{f(f^{-1}(B))}(v) &= \sup_{u \in f^{-1}(\{v\})} \mu_{f^{-1}(B)}(u) \\ &= \sup_{u \in f^{-1}(\{v\})} \mu_B(f(u)) \\ &= \mu_B(v)\end{aligned}$$

$$\begin{aligned}\sigma_{f(f^{-1}(B))}(v) &= \sup_{u \in f^{-1}(\{v\})} \sigma_{f^{-1}(B)}(u) \\ &= \sup_{u \in f^{-1}(\{v\})} \sigma_B(f(u)) \\ &= \sigma_B(v)\end{aligned}$$

$$\begin{aligned}\omega_{f(f^{-1}(B))}(v) &= \inf_{u \in f^{-1}(\{v\})} \omega_{f^{-1}(B)}(u) \\ &= \inf_{u \in f^{-1}(\{v\})} \omega_B(f(u)) \\ &= \omega_B(v)\end{aligned}$$

(6) Sia $\widetilde{B} = \langle \mathbb{V}, \mu_B, \sigma_B, \omega_B \rangle$, detto $\widetilde{C} = \widetilde{B}^{\complement} = \langle \mathbb{V}, \mu_C, \sigma_C, \omega_C \rangle$ con

$$\begin{aligned} \mu_C &= \underline{1} - \mu_B \\ \sigma_C &= \underline{1} - \sigma_B \\ \omega_C &= \underline{1} - \omega_B \end{aligned}$$

si ha che $\widetilde{f}^{-1}\left(\widetilde{C}\right) = \widetilde{f}^{-1}\left(\widetilde{B}^{\complement}\right) = \left\langle \mathbb{U}, \mu_{f^{-1}(C)}, \sigma_{f^{-1}(C)}, \omega_{f^{-1}(C)} \right\rangle$ dove $\forall u \in \mathbb{U}$ risulta:

$$\begin{aligned} \mu_{f^{-1}(C)}(u) &= \mu_C(f(u)) \\ &= 1 - \mu_B(f(u)) \\ &= \underline{1} - \mu_{f^{-1}(B)}(u), \end{aligned}$$

$$\begin{aligned} \sigma_{f^{-1}(C)}(u) &= \sigma_C(f(u)) \\ &= 1 - \sigma_B(f(u)) \\ &= \underline{1} - \sigma_{f^{-1}(B)}(u) \end{aligned}$$

e:

$$\begin{aligned} \omega_{f^{-1}(C)}(u) &= \omega_C(f(u)) \\ &= 1 - \omega_B(f(u)) \\ &= \underline{1} - \omega_{f^{-1}(B)}(u) \end{aligned}$$

D'altra parte poiché per la Definizione 2.2.1 si ha che $\widetilde{f}^{-1}\left(\widetilde{B}\right)^{\complement} = \langle \mathbb{U}\underline{1} - \mu_{f^{-1}(B)}, \underline{1} - \sigma_{f^{-1}(B)}, \underline{1} - \omega_{f^{-1}(B)} \rangle$ ne segue l'uguaglianza $\widetilde{f}^{-1}\left(\widetilde{B}^{\complement}\right) \equiv \widetilde{f}^{-1}\left(\widetilde{B}\right)^{\complement}$. □

Proposizione 2.3.6. [10] *Siano $f : \mathbb{U} \to \mathbb{V}$ una funzione tra due insiemi universo $\mathbb{U}$ e $\mathbb{V}$, $\widetilde{A}_i = \langle \mathbb{U}, \mu_{A_i}, \sigma_{A_i}, \omega_{A_i} \rangle$ (con $i = 1, 2$) due SVN-set su $\mathbb{U}$ e $\widetilde{B}_i = \langle \mathbb{V}, \mu_{B_i}, \sigma_{B_i}, \omega_{B_i} \rangle$ (con $i = 1, 2$) due SVN-set su $\mathbb{V}$, allora valgono le seguenti:*

(1) *se $\widetilde{A}_1 \Subset \widetilde{A}_2$ allora $\widetilde{f}\left(\widetilde{A}_1\right) \Subset \widetilde{f}\left(\widetilde{A}_2\right)$*

(2) *se $\widetilde{B}_1 \Subset \widetilde{B}_2$ allora $\widetilde{f}^{-1}\left(\widetilde{B}_1\right) \Subset \widetilde{f}^{-1}\left(\widetilde{B}_2\right)$*

Dimostrazione. (1) Per definizione, se $\widetilde{A}_1 \Subset \widetilde{A}_2$, significa che per ogni $u \in \mathbb{U}$, $\mu_{A_1}(u) \leq \mu_{A_2}(u)$, $\sigma_{A_1}(u) \leq \sigma_{A_2}(u)$ e $\omega_{A_1}(u) \leq \omega_{A_2}(u)$. Dobbiamo provare che, per ogni $v \in \mathbb{V}$, $\mu_{f(A_1)}(u) \leq \mu_{f(A_2)}(u)$, $\sigma_{f(A_1)}(u) \leq \sigma_{f(A_2)}(u)$ e $\omega_{f(A_1)}(u) \leq \omega_{f(A_2)}(u)$.

Se $f^{-1}(\{v\}) \neq \emptyset$ per la Definizione 2.3.1 si ha che

$$\mu_{f(A_1)}(v) = \sup_{u \in f^{-1}(\{v\})} \mu_{A_1}(u)$$

e

$$\mu_{f(A_2)}(v) = \sup_{u \in f^{-1}(\{v\})} \mu_{A_2}(u).$$

In generale, essendo $\mu_{A_1}(u) \leq \mu_{A_2}(u)$ per ogni $u \in \mathbb{U}$ risulta:

$$\sup_{u \in \mathbb{U}}(\mu_{A_1}(u)) \leq \sup_{u \in \mathbb{U}} \mu_{A_2}(u)$$

e quindi ne segue che:

$$\sup_{u\in f^{-1}(\{v\})} \mu_{A_1}(u) \leq \sup_{u\in f^{-1}(\{v\})} \mu_{A_2}(u)$$

ovvero che:

$$\mu_{f(A_1)}(u)(v) \leq \mu_{f(A_2)}(u)(v)$$

Analogamente, essendo $\sigma_{A_1}(u) \leq \sigma_{A_2}(u)$ per ogni $u \in \mathbb{U}$, si ha che

$$\sup_{u\in f^{-1}(\{v\})} \sigma_{A_1}(u) \leq \sup_{u\in f^{-1}(\{v\})} \sigma_{A_2}(u)$$

cioè:

$$\sigma_{f(A_1)}(u)(v) \leq \sigma_{f(A_2)}(u)(v)$$

Inoltre, essendo $\omega_{A_1}(u) \geq \omega_{A_2}(u)$ per ogni $u \in \mathbb{U}$

$$\inf_{u\in\mathbb{U}} \omega_{A_1}(u) \geq \inf_{u\in\mathbb{U}} \omega_{A_2}(u)$$

e quindi

$$\inf_{u\in f^{-1}(\{v\})} \omega_{A_1}(u) \geq \inf_{u\in f^{-1}(\{v\})} \omega_{A_2}(u)$$

cosicché

$$\omega_{f(A_1)}(u)(v) \geq \omega_{f(A_2)}(u)(v)$$

Se invece risulta $f^{-1}(\{v\}) = \emptyset$, si ha banalmente $1 \leq 1$ e la dimostrazione risulta completamente verificata.

(2) Per definizione, se $\widetilde{B}_1 \Subset \widetilde{B}_2$, significa che per ogni $v \in \mathbb{V}$, risulta $\mu_{B_1}(v) \leq \mu_{B_2}(v)$, $\sigma_{B_1}(v) \leq \sigma_{B_2}(v)$ e $\omega_{B_1}(v) \leq \omega_{B_2}(v)$. Dobbiamo provare che, per ogni $u \in \mathbb{U}$

$$\mu_{f^{-1}(B_1)}(u) \leq \mu_{f^{-1}(B_2)}(u),$$

$$\sigma_{f^{-1}(B_1)}(u) \leq \sigma_{f^{-1}(B_2)}(u)$$

e

$$\omega_{f^{-1}(B_1)}(u)(u) \leq \omega_{f^{-1}(B_2)}(u)(u)$$

Ma, essendo $\mu_{f^{-1}(B_1)}(u) = \mu_{B_1}(f(u))$, devo provare che $\mu_{B_1}(f(u)) \leq \mu_{B_2}(f(u))$ che è evidente poiché $f(u) \in V$ e per ipotesi sappiamo che $\mu_{B_1}(u) \leq \mu_{B_2}(u)$. Con lo stesso ragionamento, si prova che $\sigma_{B_1}(f(u)) \leq \sigma_{B_2}(f(u))$ e $\omega_{B_1}(f(u)) \geq \omega_{B_2}(f(u))$. Resta perciò provata l'inclusione $\widetilde{f}^{-1}\left(\widetilde{B}_1\right) \Subset \widetilde{f}^{-1}\left(\widetilde{B}_2\right)$.

□

Proposizione 2.3.7. [10] *Siano $f : \mathbb{U} \to \mathbb{V}$ una funzione tra due insiemi universo $\mathbb{U}$ e $\mathbb{V}$, $\left\{\widetilde{A}_i\right\}_{i\in I}$ una famiglia di SVN-set $\widetilde{A}_i = \langle \mathbb{U}, \mu_{A_i}, \sigma_{A_i}, \omega_{A_i}\rangle$ su $\mathbb{U}$ e $\left\{\widetilde{B}_i\right\}_{i\in I}$ una famiglia di SVN-set $\widetilde{B}_i = \langle \mathbb{V}, \mu_{B_i}, \sigma_{B_i}, \omega_{B_i}\rangle$ su $\mathbb{V}$, allora valgono le seguenti:*

(1) $\widetilde{f}\left(\bigcup_{i\in I} \widetilde{A}_i\right) \equiv \bigcup_{i\in I} \widetilde{f}\left(\widetilde{A}_i\right)$

(2) $\widetilde{f}\left(\bigcap_{i\in I} \widetilde{A}_i\right) \Subset \bigcap_{i\in I} \widetilde{f}\left(\widetilde{A}_i\right)$ *e vale l'uguaglianza se $\widetilde{f}$ è iniettiva*

(3) $\widetilde{f}^{-1}\left(\bigcup_{i\in I} \widetilde{B}_i\right) \equiv \bigcup_{i\in I} \widetilde{f}^{-1}\left(\widetilde{B}_i\right)$

(4) $\widetilde{f}^{-1}\left(\bigcap_{i\in I}\widetilde{B}_i\right) \equiv \bigcap_{i\in I}\widetilde{f}^{-1}\left(\widetilde{B}_i\right)$

Dimostrazione. (1) Sia $\widetilde{A} = \bigcup_{i\in I}\widetilde{A}_i$ con $\mu_A = \sup_{i\in I}\mu_{A_i}$, $\sigma_A = \sup_{i\in I}\sigma_{A_i}$ e $\omega_A = \inf_{i\in I}\omega_{A_i}$ e consideriamo $\widetilde{f}(\widetilde{A}) = \left\langle \mathbb{V}, \mu_{f(A)}, \sigma_{f(A)}, \omega_{f(A)}\right\rangle$. Inoltre, detti $\widetilde{f}(A_i) = \left\langle \mathbb{V}, \mu_{f(A_i)}, \sigma_{f(a_i)}, \omega_{f(A_i)}\right\rangle$ e $\bigcup_{i\in I}\widetilde{f}\left(\widetilde{A}_i\right) = \widetilde{F} = \langle \mathbb{V}, \mu_F, \sigma_F, \omega_F\rangle$ con $\mu_F = \sup_{i\in I}\mu_{f(A_i)}$, $\sigma_F = \sup_{i\in I}\sigma_{f(A_i)}$ e $\omega_F = \inf_{i\in I}\omega_{f(A_i)}$. Allora $\forall v \in \mathbb{V}$ si ha che:

$$\mu_{f(\sup_{i\in I}\mu_{f(A_i)})}(v) = \begin{cases} \sup\limits_{u\in f^{-1}(\{v\})}\left\{\sup\limits_{i\in I}\mu_{A_i}(u)\right\} & \text{se } f^{-1}(\{v\}) \neq \emptyset \\ 1 & \text{altrimenti} \end{cases},$$

$$\sup_{i\in I}\mu_{f(A_i)}(v) = \sup_{i\in I}\begin{cases} \sup\limits_{u\in f^{-1}(\{v\})}\mu_{A_i}(u) & \text{se } f^{-1}(\{v\}) \neq \emptyset \\ 1 & \text{altrimenti} \end{cases}$$

$$\sigma_{f(\sup_{i\in I}\sigma_{f(A_i)})}(v) = \begin{cases} \sup\limits_{u\in f^{-1}(\{v\})}\left\{\sup\limits_{i\in I}\sigma_{A_i}(u)\right\} & \text{se } f^{-1}(\{v\}) \neq \emptyset \\ 1 & \text{altrimenti} \end{cases},$$

$$\sup_{i\in I}\sigma_{f(A_i)}(v) = \sup_{i\in I}\begin{cases} \sup\limits_{u\in f^{-1}(\{v\})}\sigma_{A_i}(u) & \text{se } f^{-1}(\{v\}) \neq \emptyset \\ 1 & \text{altrimenti} \end{cases}$$

$$\omega_{f(\inf_{i\in I}\omega_{f(A_i)})}(v) = \begin{cases} \inf\limits_{u\in f^{-1}(\{v\})}\left\{\inf\limits_{i\in I}\omega_{A_i}(u)\right\} & \text{se } f^{-1}(\{v\}) \neq \emptyset \\ 1 & \text{altrimenti} \end{cases},$$

$$\inf_{i\in I}\omega_{f(A_i)}(v) = \inf_{i\in I}\begin{cases} \inf\limits_{u\in f^{-1}(\{v\})}\omega_{A_i}(u) & \text{se } f^{-1}(\{v\}) \neq \emptyset \\ 1 & \text{altrimenti} \end{cases}$$

Quindi, considerando il caso in cui $f^{-1}(\{v\}) \neq \emptyset$, si ha che:

$$\begin{aligned} \mu_{f(\sup_{i\in I}\mu_{f(A_i)})}(v) &= \sup_{u\in f^{-1}(\{v\})}\left\{\sup_{i\in I}\mu_{A_i}(u)\right\} \\ &= \sup_{u\in f^{-1}(\{v\})}\left\{\sup_{i\in I}\mu_{A_i}(u)\right\} \\ &= \sup_{i\in I}\left\{\sup_{u\in f^{-1}(\{v\})}\mu_{A_i}(u)\right\} \\ &= \sup_{i\in I}\left\{\sup_{u\in f^{-1}(\{v\})}\mu_{A_i}(u)\right\} \\ &= \sup_{i\in I}\mu_{f(A_i)}(v) \end{aligned}$$

$$\begin{aligned}
\sigma_{f(\sup_{i\in I}\sigma_{f(A_i)})}(v) &= \sup_{u\in f^{-1}(\{v\})}\left\{\sup_{i\in I}\sigma_{A_i}(u)\right\}\\
&= \sup_{u\in f^{-1}(\{v\})}\left\{\sup_{i\in I}\sigma_{A_i}(u)\right\}\\
&= \sup_{i\in I}\left\{\sup_{u\in f^{-1}(\{v\})}\sigma_{A_i}(u)\right\}\\
&= \sup_{i\in I}\left\{\sup_{u\in f^{-1}(\{v\})}\sigma_{A_i}(u)\right\}\\
&= \sup_{i\in I}\sigma_{f(A_i)}(v)
\end{aligned}$$

$$\begin{aligned}
\omega_{f(\inf_{i\in I}\mu_{f(A_i)})}(v)) &= \inf_{u\in f^{-1}(\{v\})}\left\{\inf_{i\in I}\omega_{A_i}(u)\right\}\\
&= \inf_{u\in f^{-1}(\{v\})}\left\{\inf_{i\in I}\omega_{A_i}(u)\right\}\\
&= \inf_{i\in I}\left\{\inf_{u\in f^{-1}(\{v\})}\omega_{A_i}(u)\right\}\\
&= \inf_{i\in I}\left\{\inf_{u\in f^{-1}(\{v\})}\omega_{A_i}(u)\right\}\\
&= \inf_{i\in I}\omega_{f(A_i)}(v)
\end{aligned}$$

e dunque l'uguaglianza $\widetilde{f}\left(\bigcup_{i\in I}\widetilde{A}_i\right) \equiv \bigcup_{i\in I}\widetilde{f}\left(\widetilde{A}_i\right)$.

(2) Sia $\widetilde{A} = \bigcap_{i\in I}\widetilde{A}_i$ con $\mu_A = \inf_{i\in I}\mu_{A_i}$, $\sigma_A = \inf_{i\in I}\sigma_{A_i}$ e $\omega_A = \sup_{i\in I}\omega_{A_i}$ e consideriamo $\widetilde{f}(\widetilde{A}) = \left\langle \mathbb{V}, \mu_{f(A)}, \sigma_{f(A)}, \omega_{f(A)}\right\rangle$. Inoltre, detti $\widetilde{f}(A_i) = \left\langle \mathbb{V}, \mu_{f(A_i)}, \sigma_{f(A_i)}, \omega_{f(A_i)}\right\rangle$ e $\bigcap_{i\in I}\widetilde{f}\left(\widetilde{A}_i\right) = \widetilde{F} = \langle \mathbb{V}, \mu_F, \sigma_F, \omega_F\rangle$ con $\mu_F = \inf_{i\in I}\mu_{f(A_i)}$, $\sigma_F = \inf_{i\in I}\sigma_{f(A_i)}$ e $\omega_F = \sup_{i\in I}\omega_{f(A_i)}$. Allora $\forall v \in \mathbb{V}$ si ha che:

$$\mu_{f(\inf_{i\in I}\mu_{f(A_i)})}(v) = \begin{cases} \sup\limits_{u\in f^{-1}(\{v\})}\left\{\inf\limits_{i\in I}\mu_{A_i}(u)\right\} & \text{se } f^{-1}(\{v\}) \neq \emptyset \\ 1 & \text{altrimenti} \end{cases},$$

$$\inf_{i\in I}\mu_{f(A_i)}(v) = \inf_{i\in I}\begin{cases} \sup\limits_{u\in f^{-1}(\{v\})}\mu_{A_i}(u) & \text{se } f^{-1}(\{v\}) \neq \emptyset \\ 1 & \text{altrimenti} \end{cases}$$

$$\sigma_{f(\inf_{i\in I}\sigma_{f(A_i)})}(v) = \begin{cases} \sup\limits_{u\in f^{-1}(\{v\})}\left\{\inf\limits_{i\in I}\sigma_{A_i}(u)\right\} & \text{se } f^{-1}(\{v\}) \neq \emptyset \\ 1 & \text{altrimenti} \end{cases},$$

$$\inf_{i\in I}\sigma_{f(A_i)}(v) = \inf_{i\in I}\begin{cases} \sup\limits_{u\in f^{-1}(\{v\})}\sigma_{A_i}(u) & \text{se } f^{-1}(\{v\}) \neq \emptyset \\ 1 & \text{altrimenti} \end{cases}$$

$$\omega_{f(\sup_{i\in I}\omega_{f(A_i)})}(v) = \begin{cases} \inf\limits_{u\in f^{-1}(\{v\})} \left\{\sup\limits_{i\in I}\omega_{A_i}(u)\right\} & \text{se } f^{-1}(\{v\}) \neq \emptyset \\ 1 & \text{altrimenti} \end{cases},$$

$$\sup_{i\in I}\omega_{f(A_i)}(v) \quad = \sup_{i\in I} \begin{cases} \inf\limits_{u\in f^{-1}(\{v\})} \omega_{A_i}(u) & \text{se } f^{-1}(\{v\}) \neq \emptyset \\ 1 & \text{altrimenti} \end{cases}$$

Quindi, considerando il caso in cui $f^{-1}(\{v\}) \neq \emptyset$, si ha che:

$$\begin{aligned} \mu_{f(\inf_{i\in I}\mu_{f(A_i)})}(v) &= \sup_{u\in f^{-1}(\{v\})} \left\{\inf_{i\in I}\mu_{A_i}(u)\right\} \\ &\leq \inf_{i\in I}\left\{\sup_{u\in f^{-1}(\{v\})}\mu_{A_i}(u)\right\} \\ &= \inf_{i\in I}\mu_{f(A_i)}(v) \end{aligned}$$

$$\begin{aligned} \sigma_{f(\inf_{i\in I}\sigma_{f(A_i)})}(v) &= \sup_{u\in f^{-1}(\{v\})} \left\{\inf_{i\in I}\sigma_{A_i}(u)\right\} \\ &\leq \inf_{i\in I}\left\{\sup_{u\in f^{-1}(\{v\})}\sigma_{A_i}(u)\right\} \\ &= \inf_{i\in I}\sigma_{f(A_i)}(v) \end{aligned}$$

$$\begin{aligned} \omega_{f(\sup_{i\in I}\mu_{f(A_i)})}(v)) &= \inf_{u\in f^{-1}(\{v\})} \left\{\sup_{i\in I}\omega_{A_i}(u)\right\} \\ &\geq \sup_{i\in I}\left\{\inf_{u\in f^{-1}(\{v\})}\omega_{A_i}(u)\right\} \\ &= \sup_{i\in I}\omega_{f(A_i)}(v) \end{aligned}$$

e dunque l'uguaglianza $\widetilde{f}\left(\bigcap_{i\in I}\widetilde{A}_i\right) \Subset \bigcap_{i\in I}\widetilde{f}\left(\widetilde{A}_i\right)$.

(3) Sia $\bigcup_{i\in I}\widetilde{B}_i = \widetilde{B} = \langle \mathbb{V}, \mu_B, \sigma_B, \omega_B\rangle$ con

$$\begin{aligned} \mu_B &= \sup_{i\in I}\mu_{B_i} \\ \sigma_B &= \sup_{i\in I}\sigma_{B_i} \\ \omega_B &= \inf_{i\in I}\omega_{B_i} \end{aligned}$$

Inoltre $\widetilde{f}^{-1}(\widetilde{B}) = \left\langle \mathbb{U}, \mu_{f^{-1}(B)}, \sigma_{f^{-1}(B)}, \omega_{f^{-1}(B)}\right\rangle$ con

$$\begin{aligned} \mu_{f^{-1}(B)}(u)(u) &= \mu_B(f(u)) \\ \sigma_{f^{-1}(B)}(u)(u) &= \sigma_B(f(u)) \\ \omega_{f^{-1}(B)}(u)(u) &= \omega_B(f(u)) \end{aligned}$$

Ma

$$\begin{aligned}\mu_B(f(u)) &= \sup_{i\in I} \mu_{B_i}(f(u)) \\ &= \sup_{i\in I} f^{-1}(\mu_{B_i})(u)\end{aligned}$$

$$\begin{aligned}\sigma_B(f(u)) &= \sup_{i\in I} \sigma_{B_i}(f(u)) \\ &= \sup_{i\in I} f^{-1}(\sigma_{B_i})(u)\end{aligned}$$

$$\begin{aligned}\omega_B(f(u)) &= \inf_{i\in I} \omega_{B_i}(f(u)) \\ &= \inf_{i\in I} f^{-1}(\omega_{B_i})(u)\end{aligned}$$

da qui ne discende la tesi.

(4) Sia $\bigcap_{i\in I} \widetilde{B}_i = \widetilde{B} = \langle \mathbb{V}, \mu_B, \sigma_B, \omega_B \rangle$ con

$$\begin{aligned}\mu_B &= \inf_{i\in I} \mu_{B_i} \\ \sigma_B &= \inf_{i\in I} \sigma_{B_i} \\ \omega_B &= \bigvee_{i\in I} \omega_{B_i}\end{aligned}$$

Inoltre $\widetilde{f}^{-1}(\widetilde{B}) = \left\langle \mathbb{U}, \mu_{f^{-1}(B)}, \sigma_{f^{-1}(B)}, \omega_{f^{-1}(B)} \right\rangle$ con

$$\begin{aligned}\mu_{f^{-1}(B)}(u) &= \mu_B(f(u)) \\ \sigma_{f^{-1}(B)}(u) &= \sigma_B(f(u) \\ \omega_{f^{-1}(B)}(u) &= \omega_B(f(u)\end{aligned}$$

Ma

$$\begin{aligned}\mu_B(f(u)) &= \inf_{i\in I} \mu_{B_i}(f(u)) \\ &= \inf_{i\in I} \mu_{f^{-1}(B_i)}(u)\end{aligned}$$

$$\begin{aligned}\sigma_B(f(u)) &= \inf_{i\in I} \sigma_{B_i}(f(u)) \\ &= \inf_{i\in I} \sigma_{f^{-1}(B_i)}(u)\end{aligned}$$

$$\begin{aligned}\omega_B(f(u)) &= \sup_{i\in I} \omega_{B_i}(f(u)) \\ &= \sup_{i\in I} \omega_{f^{-1}(B_i)}(u)\end{aligned}$$

e ne discende quindi la tesi.

□

Capitolo 3

Morfologia Matematica

In questo capitolo, esploreremo le fondamenta delle immagini digitali, focalizzandoci sulle rappresentazioni discrete delle informazioni visive e sui concetti di immagini a livelli di grigio e binarie. Successivamente, introdurremo la Morfologia Matematica e ne studieremo le operazioni elementari che permettono di analizzare, manipolare e comprendere la struttura e la forma delle immagini.

3.1 Immagini Digitali

Le immagini digitali sono l'oggetto principale di questa tesi. Esse sono prodotte da una grande varietà di dispositivi fisici quali fotocamere, scanner, macchinari per i raggi X, microscopi elettronici, etc. e vengono usate per molteplici scopi: dalla diagnostica medica, al rilevamento di dati geografici per usi statistici o militari, per il controllo di qualità nelle industrie, fino al campo dell'intrattenimento [3].

Un'immagine digitale può essere definita come una rappresentazione numerica di un'immagine bidimensionale o tridimensionale attraverso una griglia discreta di elementi, chiamati *pixel* (abbreviazione della locuzione inglese *picture element*). Ogni pixel contiene informazioni riguardo al colore, all'intensità luminosa o ad altre caratteristiche visive dell'immagine in una determinata posizione spaziale.

Formalmente, un'***immagine digitale*** bidimensionale (o 2D) può essere rappresentata come una funzione $I : \mathbb{Z}^2 \to \mathbb{R}$, dove per ogni $(x, y) \in \mathbb{Z}^2$, $I(x, y)$ restituisce il valore numerico (detto livello di grigio o intensità) corrispondente al pixel di coordinate (x, y) nell'immagine.

Le immagini digitali possono essere classificate in diverse categorie in base al numero di valori che i pixel possono assumere. Due casi particolari di particolare rilevanza sono le immagini a livelli di grigio e le immagini binarie.

Nelle immagini a livelli di grigio, ogni pixel ha associato un valore che rappresenta l'intensità luminosa o la tonalità di grigio del punto corrispondente nell'immagine. I valori tipicamente variano da 0 (corrispondente al nero assoluto) fino al valore massimo $2^n - 1$ (che codifica il bianco assoluto), dove n è il numero di bit utilizzati per la rappresentazione del colore. Il codominio T della funzione immagine non è quindi l'insieme continuo $\mathbb{R}$ dei numeri reali ma un sottoinsieme finito e discreto dell'insieme $\mathbb{Z}$ dei numeri interi relativi o addirittura dell'insieme $\mathbb{N}$ dei numeri naturali. Nell maggior parte dei casi si utilizzano immagini a 8 bit in cui il valore massimo $2^8 - 1 = 255$ corrisponde al bianco assoluto .

La rappresentazione matematica di un'immagine a livelli di grigio ad n bit [4,14,18] può essere quindi considerata come una funzione $I : \mathbb{Z}^2 \to T$ dove $T = \{0, 1, 2, \ldots 2^n - 1\}$ e $I(x, y)$ rappresenta il valore del livello di grigio del pixel di coordinate intere (x, y). In

particolare, per immagini digitali a 8 bit il codominio dell'immagine $I : \mathbb{Z}^2 \to T$ sarà l'insieme $T = \{0, 1, 2, \ldots 255\}$. La Figura 3.1 mostra come una immagine digitale a livelli di grigio viene rappresentata matematicamente mediante una tabella discreta di valori interi.

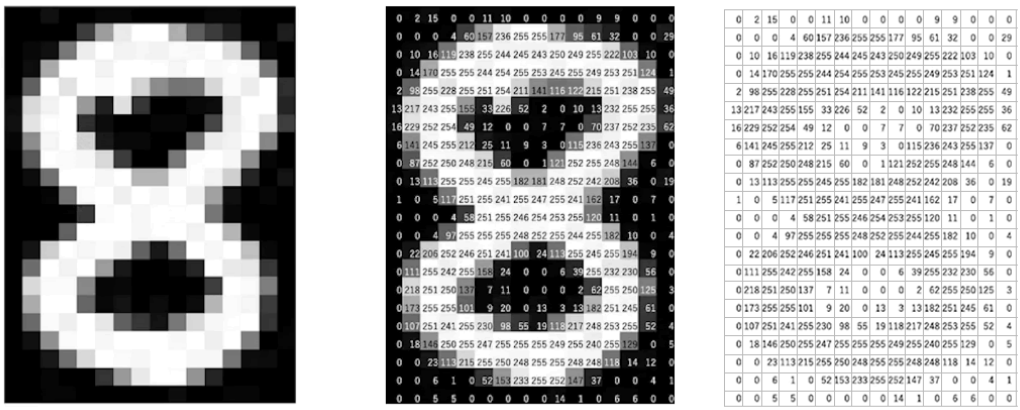
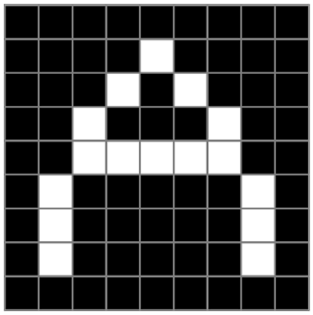


Figura 3.1: Esempio di codifica di una immagine digitale a livelli di grigio.

Le immagini binarie (comunemente dette in bianco e nero) sono un caso speciale di immagini a toni di grigio in cui le informazioni di colore sono codificate con un singolo bit, ossia quando $n = 1$ e il codominio dell'immagine $I : \mathbb{Z}^2 \to T$ si riduce semplicemente ai due valori $T = \{0, 1\}$ dove 0 continua a rappresentare il nero assoluto e 1 corrisponde per convenzione al bianco assoluto.
Data una immagine binaria, l'insieme di tutti i pixel accesi (generalmente di colore bianco nella rappresentazione sullo schermo) prende il nome di immagine in primo piano (in inglese *foreground*), mentre l'insieme di tutti i pixel spenti (generalmente di colore nero sullo schermo) prende il nome di sfondo (in inglese *background*).
In effetti, la decisione di considerare i pixel neri come accesi e bianchi come spenti è del tutto arbitraria e rappresenta una convenzione che si mantiene per ragioni storiche (poiché agli albori dell'Analisi delle Immagini, quando il colore di sfondo dei primi semplici monitor catodici era il nero e l'immagine veniva formata sullo schermo illuminando elettronicamente alcuni fosfori verdi o bianchi corrispondenti ai pixel) ma la definizione può facilmente essere invertita, ossia assumendo come immagine in primo piano quella corrispondente ai pixel accesi di colore nero, come succede ad esempio considerando immagini tracciate in nero su uno sfondo bianco, che è la situazione più tipica negli odierni applicativi di computer grafica (quali Photoshop, Gimp, CorelDraw, ecc.).

In ogni caso, secondo le convenzioni più comuni, data una immagine binaria $I : \mathbb{Z}^2 \to \{0, 1\}$, la corrispondente immagine in primo piano è rappresentato dalla fibra del valore 1 mediante l'immagine I, ossia $P(I) = I^{-1}(\{1\}) = \{(x, y) \in \mathbb{Z}^2 : I(x, y) = 1\}$ ed è quindi un sottoinsieme discreto dello spazio euclideo bidimensionale che costituisce una descrizione completa dell'immagine binaria. La Figura 3.2 mostra come una immagine digitale a livelli di grigio viene rappresentata matematicamente mediante una tabella discreta di valori interi.

Vale la pena osservare che esiste una corrispondenza biunivoca tra le immagini binarie intese come funzioni $I : \mathbb{Z}^2 \to \{0, 1\}$ e le loro immagini in primo piano $P(I)$ e per questo motivo, anche nel seguito, ci riferiremo spesso alle immagini binarie come semplici sottoinsiemi del piano discreto $\mathbb{Z}^2$.



0	0	0	0	0	0	0	0	0
0	0	0	0	1	0	0	0	0
0	0	0	1	0	1	0	0	0
0	0	1	0	0	0	1	0	0
0	0	1	1	1	1	1	0	0
0	1	0	0	0	0	0	1	0
0	1	0	0	0	0	0	1	0
0	1	0	0	0	0	0	1	0
0	0	0	0	0	0	0	0	0

0	0	0	0	0	0	0	0	0
0	0	0	0	1	0	0	0	0
0	0	0	1	0	1	0	0	0
0	0	1	0	0	0	1	0	0
0	0	1	1	1	1	1	0	0
0	1	0	0	0	0	0	1	0
0	1	0	0	0	0	0	1	0
0	1	0	0	0	0	0	1	0
0	0	0	0	0	0	0	0	0

Figura 3.2: Esempio di codifica di una immagine binaria.

Inoltre, nelle applicazioni pratiche il dominio di una immagine si riduce ad un opportuno sottoinsieme discreto di $\mathbb{Z}^2$ o addirittura di $\mathbb{N}^2$ ossia ad una griglia finita di interi che rappresentano le coordinate di ciascun pixel e, generalmente nella rappresentazione grafica, si assume che il sistema di riferimento abbia origine nell'angolo in alto a sinistra, che l'asse X delle ascisse sia orientato verso destra e che invece l'asse Y sia orientato verso il basso. Di conseguenza nelle applicazioni, le immagini a livelli di grigio e le immagini binarie si riducono ad una funzione:

$$I : [x_{min}, x_{max}]_{\mathbb{Z}} \times [y_{min}, y_{max}]_{\mathbb{Z}} \to T$$

che ad ogni pixel di coordinate (x, y) fa corrispondere il suo valore di intensità discreto $I(x, y)$ ed in cui:

- l'intervallo discreto $[x_{min}, x_{max}]_{\mathbb{Z}} = \{x \in \mathbb{Z} : x_{min} \leq x \leq x_{max}\}$ è l'insieme dei numeri relativi delle ascisse compresi tra il valore minimo x_{min} e il valore massimo x_{max},
- l'intervallo discreto $[y_{min}, y_{max}]_{\mathbb{Z}} = \{y \in \mathbb{Z} : y_{min} \leq y \leq y_{max}\}$ è l'insieme dei numeri relativi delle ordinate compresi tra il valore minimo y_{min} e il valore massimo y_{max}, e
- il codominio $T = \{0, 1, 2, \ldots 2^n - 1\}$ rappresenta l'insieme dei livelli di grigio dell'immagine, generalmente $T = \{0, \ldots 255\}$ per le immagini a 8 bit (con 256 livelli di grigio) o $T = \{0, 1\}$ per le immagini binarie.

In questo caso diremo ***larghezza*** w_I ed ***altezza*** h_I dell'immagine digitale I rispettivamente le ampiezze dell'intervallo discreto delle ascisse e delle ordinate, ossia porremo $w_I = x_{max} - x_{min} + 1$ ed $h_I = y_{max} - y_{min} + 1$.

Il complementare $A^{\complement}$ di una immagine binaria $A \subseteq \mathbb{Z}^2$ è evidentemente l'immagine binaria in cui i pixel bianchi sono scambiati con i pixel neri e viceversa.

Si parla di elaborazione di immagini per intendere il procedimento che consiste nell'applicare ad un'immagine digitalizzata un certo operatore che la trasformi in una versione modificata della stessa allo scopo di facilitarne l'interpretazione e di estrarne informazioni. Le trasformazioni delle immagini sono di tre tipi: unarie (un'immagine in entrata ed una in uscita), diadiche (due immagini in entrata ed una in uscita) e di estrazione dell'informazione (immagine in entrata ed informazione in uscita).

3.2 Morfologia Binaria

In generale, un operatore morfologico trasforma un'immagine in un'altra immagine attraverso l'utilizzo di una ulteriore immagine. L'idea essenziale è quella di esplorare un'immagine attraverso una forma predefinita, detta *elemento strutturante* e che generalmente è di piccole dimensioni rispetto all'immagine principale. Esempi di operatori morfologici elementari (o di base) che costituiscono le fondamenta di questa disciplina sono la dilatazione, l'erosione, l'apertura e la chiusura.

Definizione 3.2.1. [3] *Dato un insieme discreto $A \subseteq \mathbb{Z}^2$, diciamo **riflessione** (o simmetrico) di A e lo indichiamo con $\breve{A}$, l'insieme di tutti i punti opposti di A, cioé:*

$$\breve{A} = \{-a,\ a \in A\}$$

Osservazione 3.2.2. *Chiaramente la riflessione è una involuzione in quanto, applicata due volte di seguito, riporta all'insieme di partenza, cioè $\breve{\breve{A}} = A$. Infatti per ogni $x \in A$, si può scrivere che $x = -(-x)$, con $-x \in \breve{A}$ e dunque $-(-x) \in \breve{\breve{A}}$. Ciò implica che $A \subseteq \breve{\breve{A}}$. Viceversa, se $x \in \breve{\breve{A}}$, ciò vuol dire che $x = -a$ con $a \in \breve{A}$ e ciò implica che $a = -b$ con $b \in A$. Allora $x = -a = -(-b) = b$ con $b \in A$. Da qui si ha che $\breve{\breve{A}} \subseteq A$.*

Inoltre vale la seguente proprietà che lega la riflessione al complementare.

Proprietà 1. *Dato un insieme discreto $A \subseteq \mathbb{Z}^2$, il complementare della riflessione di B coincide con la riflessione del suo complementare, cioè:*

$$(\breve{A})^{\complement} = \breve{(A^{\complement})}$$

Dimostrazione. Infatti, dire che $x \in (\breve{A})^{\complement}$ equivale a dire che $x \notin \breve{A}$ ossia che $-x \notin A$ (perché altrimenti $x \in \breve{A}$) e dunque che $-x \in A^{\complement}$ che equivale a dire che $x \in \breve{(A^{\complement})}$. □

Definizione 3.2.3. [3] *Sia $A \subset \mathbb{Z}^2$ e $x \in \mathbb{Z}^2$. Il traslato di A tramite x, denotato con A_x è definito come*

$$A_x = \{a + x : a \in A\}$$

Proprietà 2. *Sia $A \subseteq \mathbb{Z}^2$ e $x \in \mathbb{Z}^2$. Se $y \in A_x$ allora $y - x \in A$.*

Dimostrazione. Se $y \in A_x$ si ha che esiste qualche $a \in A$ tale che $y = a + x$ e quindi anche che $y - x = a \in A$. □

Proprietà 3. *Dato un insieme discreto $A \subseteq \mathbb{Z}^2$, il complementare del traslato coincide con il traslato del complementare, cioé:*

$$(A_x)^{\complement} = (A^{\complement})_x$$

Dimostrazione. Per ogni $y \in (A^{\complement})_x$ si ha che esiste qualche $c \in A^{\complement}$ tale che $y = c + x$. Allora $y \notin A_x$ in quanto se – per assurdo – esistesse qualche $a \in A$ tale che $y = a + x$ ne seguirebbe che $c + x = y = a + x$ e quindi che $c = a \in A$ contro il fatto che $c \in A^{\complement}$ ossia che $c \notin A$. Ciò prova che $y \notin A_x$ ovvero che $y \in (A_x)^{\complement}$ e dunque che vale l'inclusione $(A^{\complement})_x \subseteq (A_x)^{\complement}$.

Viceversa, per ogni $y \in (A_x)^{\complement}$, ossia tale che $y \notin A_x$ si ha che $y \in (A^{\complement})_x$. Infatti, se per assurdo, fosse $y \notin (A^{\complement})_x$, potremmo considerare il punto $z = y - x$ ed osservare che deve necessariamente essere $z \notin A$ (perché altrimenti se fosse $z \in A$ si avrebbe $y = z + x \in A_x$ contro il fatto che $y \notin A_x$). D'altronde, dal fatto che $y \notin (A^{\complement})_x$ ne segue pure che $z \notin A^{\complement}$ (perché altrimenti si avrebbe $y = z + x \in (A^{\complement})_x$ contro l'ipotesi che $y \notin (A^{\complement})_x$) e dunque che $z \in A$ contraddicendo il fatto che $z \notin A$. Ciò prova che, per ogni $y \in (A_x)^{\complement}$, si ha $y \in (A^{\complement})_x$ ovvero che $(A_x)^{\complement} \subseteq (A^{\complement})_x$ e dunque l'uguaglianza. □

La Figura 3.3 illustra la formula della Proprietà 3.

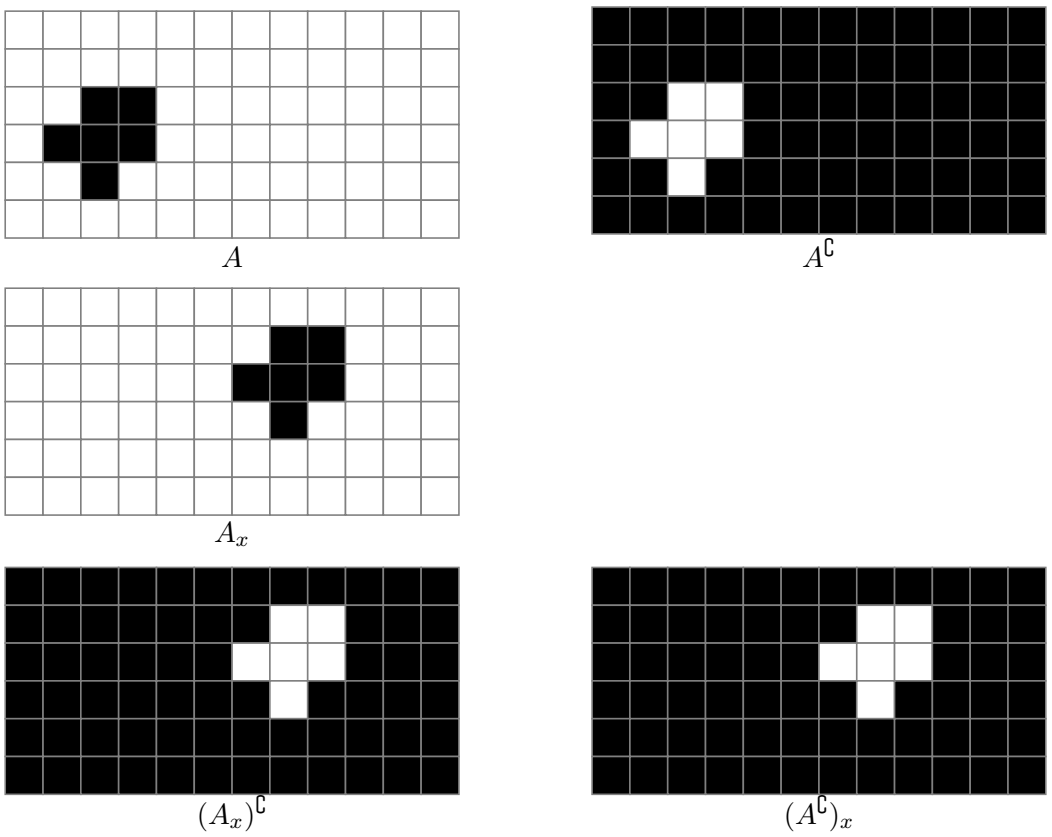


Figura 3.3: Il Complementare $(A_x)^{\complement}$ di un traslato coincide col traslato $(A^{\complement})_x$ del complementare.

Gli operatori morfologici operano con due immagini, in particolare, una di esse, chiamata *elemento strutturante*. Ogni elemento strutturante ha una forma designata che può essere usata per indagare l'immagine. Dunque un'immagine può essere modificata attraverso un elemento strutturante di diverse forme e misure. Le operazioni elementari sono dilatazione ed erosione, le quali possono essere combinate in sequenza per produrre altre operazioni come *apertura* e *chiusura*.

Innanzitutto, è necessario definire la somma e la sottrazione di Minkowski.

Definizione 3.2.4. *(Somma di Minkowski) Dati due insiemi A e $B \subseteq \mathbb{Z}^2$, diciamo* ***somma di Minkowski*** *tra A e B e la indichiamo con $A + B$, la somma di tutti i punti di A e di B, cioè:*

$$A + B = \{a + b : a \in A, b \in B\}$$

Definizione 3.2.5. *(Sottrazione di Minkowski) Dati due insiemi A e $B \subseteq \mathbb{Z}^2$, diciamo **sottrazione di Minkowski** tra A e B e la indichiamo con $A - B$, la somma di Minkowski di A e della riflessione di B, cioè:*

$$A - B = A + \breve{B}$$

3.2.1 Erosione binaria

Definizione 3.2.6. [3] *Siano $A,B \subseteq \mathbb{Z}^2$ due insiemi discreti. Diciamo **erosione** di A rispetto all'elemento strutturante B e lo indichiamo con $A \ominus B$ l'insieme di tutti i punti $x \in Z^2$ in cui il traslato di B in quel punto è interamente contenuto nell'insieme A, cioè in simboli:*

$$A \ominus B = \{x \in \mathbb{Z}^2 : B_x \subseteq A\}$$

L'erosione di un'immagine consiste nel posizionare l'elemento strutturante in ogni regione dell'immagine e verificare se esso è contenuto completamente nell'immagine in quella regione. Se l'elemento strutturante è contenuto completamente nell'insieme, allora il suo punto centrale viene considerato parte del risultato dell'erosione. In caso contrario, il punto centrale dell'elemento strutturante viene rimosso dalla maschera risultante. La Figura 3.4 illustra la definizione di erosione neutrosofica.

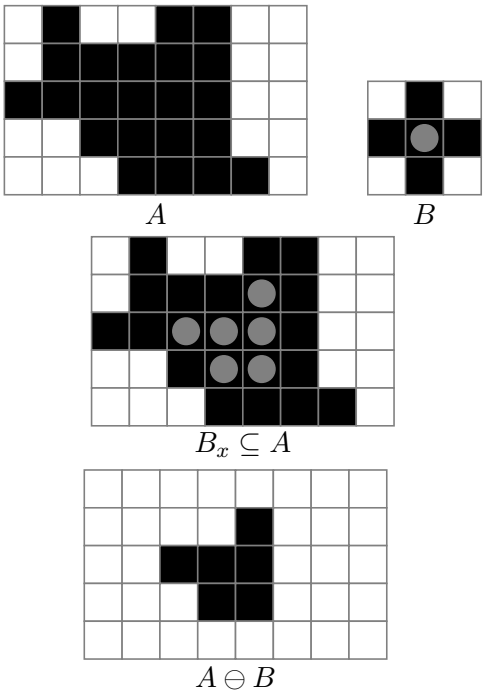


Figura 3.4: Erosione $A \ominus B$ di un insieme A rispetto a un elemento strutturante B

L'erosione può essere rappresentata anche come intersezione di immagini, come mostra la seguente proprietà.

Proprietà 4. [3] *Dati un insieme discreto A ed un elemento strutturante B in $\mathbb{Z}^2$, l'erosione $A \ominus B$ è data dall'intersezione di tutte le traslazioni di A rispetto ai punti opposti di B, in simboli*

$$A \ominus B = \bigcap_{b \in B} A_{-b}$$

Dimostrazione. Per ogni $x \in A \ominus B$ si ha che $B_x \subseteq A$. Proviamo che $x \in \bigcap_{b\in B} A_{-b}$. Infatti, per ogni $b \in B$, dal fatto che $B_x = \{b + x : b \in B\}$, si ha che $b + x \in B_x \subseteq A$. Ciò significa che esiste qualche elemento $a \in A$ tale che $x = a + (-b)$ che per definizione di traslato, $x \in A_{-b}$. Dal momento che questo vale per ogni $b \in B$, ne segue che $x \in \bigcap_{b\in B} A_{-b}$. Da qui l'inclusione $A \ominus B \subseteq \bigcap_{b\in B} A_{-b}$.
Viceversa, per ogni $x \in \bigcap_{b\in B} A_{-b}$, si ha che $x \in A_{-b}$ per ogni $b \in B$. Ciò significa che esiste qualche elemento $a \in A$ tale che $x = a + (-b)$, da cui segue che $b + x = a$ per ogni $b \in B$, cioè $B_x \subseteq A$ per definizione di traslato, cioè $x \in A \ominus B$ per definizione di erosione. Resta dunque provata la tesi.

□

Proposizione 3.2.7. [14] *Dati A, B, C* $\subseteq \mathbb{Z}^2$*, valgono le seguenti proprietà:*

(1) *Se* $0 \in B$ *allora* $A \ominus B \subseteq A$

(2) *Se* $A \subseteq B$ *allora* $A \ominus C \subseteq B \ominus C$

(3) $(A \cap B) \ominus C = (A \ominus C) \cap (B \ominus C)$

Dimostrazione. (1) Per ogni $x \in A \ominus B$ si ha che $B_x \subseteq A$ cioè che per ogni $b \in B$ risulta $b + x \in A$ e giacché per ipotesi $0 \in B$ si ha che $0 + x \in A$ ossia che $x \in A$.
(2) Sia $x \in A \ominus C$, allora $C_x \subseteq A$ per definizione di erosione. Ma, poiché per ipotesi $A \subseteq B$, si ha che $C_x \subseteq B$ e dunque $x \in B \ominus C$. Ciò prova l'asserto.
(3) Sia $x \in (A \cap B) \ominus C$, allora $C_x \subseteq A \cap B$ per definizione di erosione. Da qui segue che $C_x \subseteq A$ ed anche $C_x \subseteq B$, ma ciò vuol dire che $x \in A \ominus C$ ed anche $x \in B \ominus C$ e dunque $x \in (A \ominus C) \cap (B \ominus C)$.

□

3.2.2 Dilatazione binaria

Definizione 3.2.8. [3] *Siano A,B* $\subseteq \mathbb{Z}^2$ *due insiemi discreti. Diciamo* ***dilatazione*** *di A rispetto all'elemento strutturante B e lo indichiamo con* $A \oplus B$*, il complementare dell'erosione del complementare dell'insieme A rispetto alla riflessione dell'elemento strutturante, in simboli:*

$$A \oplus B = (A^{\complement} \ominus \breve{B})^{\complement}$$

Per dilatare A mediante B, si ruota B di 180 gradi intorno all'origine per ottenere la riflessione $\breve{B}$, si erode il complementare $A^{\complement}$ di A mediante $\breve{B}$ e si prende il complemento dell'erosione. La dilatazione dell'immagine A tramite l'elemento strutturante B si può tradurre come un'espansione dell'immagine stessa. Per comprendere meglio, supponiamo di utilizzare erosione e dilatazione per la pianificazione dei percorsi dei robot. L'elemento strutturante è la forma del robot, mentre l'immagine di input è la regione in cui il robot può muoversi (il robot non può ruotare). Se da un lato l'erosione costituisce l'insieme dei punti di tutte le coordinate del robot in cui esso stesso può muoversi, dall'altro lato la dilatazione costituisce l'insieme dei punti in cui il robot riflesso non può muoversi dall'esterno. La Figura 3.5 è un esempio per rappresentare la definizione di dilatazione binaria.

Proprietà 5. [3] *Dati un insieme discreto A ed un elemento strutturante B in* $\mathbb{Z}^2$*, la dilatazione* $A \oplus B$ *è data dall'unione di tutte le traslazioni di A rispetto ai punti di B, cioè*

$$A \oplus B = \bigcup_{b\in B} A_b$$

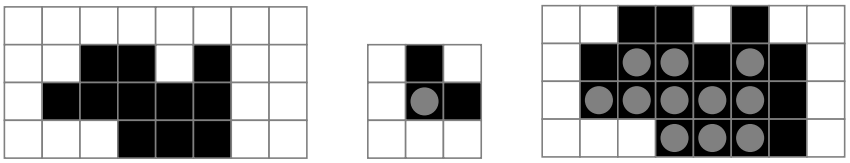


Figura 3.5: Dilatazione $A \oplus B$ di un insieme A rispetto a un elemento strutturante B

Dimostrazione. Infatti si ha che:

$$A \oplus B = \left(A^{\complement} \ominus \breve{B}\right)^{\complement}$$

(usando la Proprietà 4)

$$= \left(\bigcap_{x \in \breve{B}} \left(A^{\complement}\right)_{-x} \right)^{\complement}$$

(ricordando dalla Proprietà 3 che il traslato di un complementare coincide con il complementare del traslato 4)

$$= \left(\bigcap_{x \in \breve{B}} (A_{-x})^{\complement} \right)^{\complement}$$

(utilizzando la formula di De Morgan)

$$= \bigcup_{x \in \breve{B}} \left((A_{-x})^{\complement}\right)^{\complement}$$
$$= \bigcup_{x \in \breve{B}} A_{-x}$$

(osservando che $x = -b \in \breve{B}$ equivale a $b = -x \in B$)

$$= \bigcup_{b \in B} A_b$$

□

Osservazione 3.2.9. *Osserviamo che la dilatazione $A \oplus B$ di un insieme A rispetto ad un elemento strutturante B, coincide con la loro addizione di Minkowski $A + B$ definita in 3.2.4. Infatti:*

$$A \oplus B = \bigcup_{b \in B} A_b = \{a + b : a \in A, b \in B\} = A + B$$

Proposizione 3.2.10. [14] *Siano A, B, $C \subseteq \mathbb{Z}^2$ allora valgono le seguenti proprietà:*

(1) *Se $0 \in B$ allora $A \subseteq A \oplus B$*

(2) $A \oplus B = B \oplus A$

(3) $(A \oplus B) \oplus C = A \oplus (B \oplus C)$

(4) *Se* $A \subseteq B$ *allora* $A \oplus C \subseteq B \oplus C$

(5) $(A \cap B) \oplus C \subseteq (A \oplus C) \cap (B \oplus C)$

(6) $(A \cup B) \oplus C = (A \oplus C) \cup (B \oplus C)$

Dimostrazione. (1) Sia $x \in A$. Possiamo scrivere $x = x + 0$ con $x \in A$ e $0 \in B$. Poiché l'addizione di Minkowski coincide con la dilatazione, allora $x \in A + B = A \oplus B$, da cui l'asserto.
(2) Sia $x \in A \oplus B$, allora $x = a + b$ per qualche $a \in A$ e $b \in B$. Dal momento che la dilatazione coincide con la somma di Minkowski, $x = a + b = b + a$ con $b \in B$ e $a \in A$. Allora $x \in B \oplus A$.
(3) Sia $x \in (A \oplus B) \oplus C$, allora $x = y + c$ per qualche $y \in A \oplus B$ e $c \in C$. Ma $y \in A \oplus B$ significa che per qualche $a \in A$ e per qualche $b \in B$ (dal momento che l'addizione di Minkowski coincide con la dilatazione), $y = a + b$ e dunque $x = a + b + c$, ma $b + c \in B + C = B \oplus C$. Dunque $x = a + z$ con $a \in A$ e $z \in B \oplus C$, quindi $x \in A \oplus (B \oplus C)$.
(4) Sia $x \in A \oplus C$, allora per qualche $a \in A$ e per qualche $c \in C$, $x = a + c$. Ma $a \in A \subseteq B$, dunque $x = a + c$ con $a \in B$ e $c \in C$. Allora $x \in B \oplus C$.
(5) Sia $x \in (A \cap B) \oplus C$. Per il fatto che l'addizione di Minkowski coincide con la dilatazione, si può dire che $x = l + c$ con $l \in A \cap B$ e $c \in C$, ma $l \in A \cap B$ significa che $l \in A$ ed anche $l \in B$. Dunque $x = l + c$ con $l \in A$ e $c \in C$, ma anche $l \in B$ e $c \in C$. Quindi $x \in A \oplus C$ ed anche $x \in B \oplus C$, allora $x \in (A \oplus C) \cap (B \oplus C)$.
(6) Sia $x \in (A \cup B) \oplus C$. Per il fatto che l'addizione di Minkowski coincide con la dilatazione, si può dire che $x = t + c$ con $t \in A \cup B$ e $c \in C$, ma $l \in A \cup B$ significa che $l \in A$ o $l \in B$. Dunque $x = l + c$ con $l \in A$ e $c \in C$ oppure $l \in B$ e $c \in C$. Quindi $x \in A \oplus C$ o $x \in B \oplus C$, allora $x \in (A \oplus C) \cup (B \oplus C)$.

□

Proposizione 3.2.11. [14] *Siano* $A, B, C \subseteq \mathbb{Z}^2$ *allora valgono le seguenti proprietà che legano erosione e dilatazione:*

(1) $A \oplus (B \ominus C) \subseteq (A \oplus B) \ominus C$

(2) $(A \ominus B) \ominus C = A \ominus (B \oplus C)$

(3) $A \oplus (B \ominus C) \subseteq (A \oplus B) \ominus C$

(4) $(A \oplus B)^{\complement} = A^{\complement} \ominus \breve{B}$ *(dualità)*

(5) $A \ominus B = (A^C \oplus \breve{B})^C$

Dimostrazione. (1) Sia $x \in A \oplus (B \ominus C)$, allora, per definizione di dilatazione, $x = a + y$ con $a \in A$ e $y \in B \ominus C$. Da quest'ultima appartenenza si sa che $y \in B$ poiché $B \ominus C \subseteq B$ ed inoltre, per definizione di erosione, $C_y \subseteq B$ cioè per ogni $c \in C$, $c + y \in B$. Allora $x + c = a + y + c \in A + B$ dove $a \in A$ e $y + c \in B$. Ma poiché la somma di Minkowski coincide con la dilatazione, $x + c \in A \oplus B$, cioè $C_x \subseteq A \oplus B$, quindi $x \in (A \oplus B) \ominus C$, da cui l'asserto.
(2) Per ogni $x \in (A \ominus B) \ominus C$ si ha che $C_x \subseteq A \ominus B$ e da qui, per definizione di traslazione, ne segue che per ogni $c \in C$ risulta $c + x \in A \ominus B$ (cioè che $B_{c+x} \subseteq A$) e dunque, ancora una volta per definizione di traslazione, che per ogni $b \in B$, $b + c + x \in A$. Allora si può affermare che $x \in A \ominus (B \oplus C)$. Infatti, per ogni $y \in (B \oplus C)_x$ si ha che esiste qualche $l \in B \oplus C$ tale che $y = l + x$, ma poiché sappiamo che la dilatazione coincide con

la somma di Minkowski, ne segue che esisteranno degli elementi $b \in B$ e $c \in C$ tali che $l = b + c$ e dunque tali che $y = b + c + x \in A$ e ciò prova che $(B \oplus C) \subseteq A$ e che, infine, $x \in A \ominus (B \oplus C)$.
Viceversa, per ogni $x \in A \ominus (B \oplus C)$, per la definizione di erosione, si ha che $(B \oplus C)_x \subseteq A$ da cui, per definizione di traslazione e ricordando che la dilatazione coincide con la somma di Minkowski, si ha che per ogni coppia di elementi $b \in B$ e $c \in C$ risulta $b + c + x \in A$. Ma allora possiamo affermare che $x \in (A \ominus B) \oplus C$, ossia che $C_x \subseteq A \ominus B$. Infatti per ogni $y \in C_x$ si ha che esiste qualche $c \in C$ tale che $y = c + x$. Da qui, sapendo che $b + c + x \in A$ posso dire che $b + y = b + c + x \in A$ e cioè che $B_y \subseteq A$ e quindi che $y \in A \ominus B$ il che prova che $C_x \subseteq A \ominus B$ e quindi che $x \in (A \ominus B) \ominus C$.
(3)Sia $x \in A \oplus (B \ominus C)$. Per la Proprietà 5 della dilatazione espressa come unione di traslati, si ha che $x \in A \oplus (B \ominus C) = \bigcup_{y \in B \ominus C} A_y$ e quindi esiste qualche $y \in B \ominus C$ tale che $x \in A_y$ da cui, per la Proprietà 2, segue subito che $x - y \in A$. Per definizione di erosione, dalla relazione $y \in B \ominus C$ si deduce che $C_y \subseteq B$, cioè che per ogni $v \in C$ si ha che $v + y \in B$. Dal momento che vogliamo provare che $x \in (A \oplus B) \ominus C$, cioè (per definizione di erosione) che $C_x \subseteq A \oplus B$, consideriamo un qualsiasi $z \in C_x$ cosicché esisterà qualche $v \in C$ tale che $zz = v + x$. Da qui, aggiungendo e sottraendo il punto y e ricordando il fatto che la Somma di Minkowski coincide con la dilatazione, si ottiene che:

$$z = x + v = x - y + y + v = (x - y) + (y + v) \in A \oplus B$$

in quanto sappiamo che $x - y \in A$ e che, per $v \in C$, $v + y \in B$. Dunque resta provato che $C_x \subseteq A \oplus B$ e quindi che $x \in (A \oplus B) \ominus C$ dimostrando che vale l'inclusione $A \oplus (B \ominus C) \subseteq (A \oplus B) \ominus C$.

(4) La tesi segue direttamente dalla Definizione 3.2.8 di dilatazione $A \oplus B = \left(A^{\complement} \ominus \breve{B}\right)^{\complement}$ applicando il complementare ad ambo i membri.

(5) L'asserto discende direttamente dalla Definizione 3.2.8 applicata ad $A^{\complement} \oplus \breve{B}$ e ricordando che la riflessione è un'involuzione. Infatti, $A^{\complement} \oplus \breve{B} = \left(\left(A^{\complement}\right)^{\complement} \ominus \widetilde{\left(\breve{B}\right)}\right)^{\complement} = (A \ominus B)^{\complement}$.
Passando al complementare ambo i membri otteniamo la tesi, cioè $A \ominus B = (A^{\complement} \oplus \breve{B})^{\complement}$.
□

3.2.3 Apertura binaria

Definizione 3.2.12. [14] *Siano $A,B \in \mathbb{Z}^2$ due insiemi discreti. Diciamo **apertura** (più propriamente apertura binaria) di A rispetto all'elemento strutturante B e la indichiamo con $A \circ B$, la dilatazione dell'erosione di A rispetto allo stesso elemento strutturante B, cioè*

$$A \circ B = (A \ominus B) \oplus B$$

La Figura 3.7 illustra la definizione di apertura binaria.

Osservazione 3.2.13. *Generalmente, l'apertura può essere usata per rimuovere piccoli oggetti e connessioni tra oggetti.*

Proposizione 3.2.14. [3, 14] *Siano A, B, $C \subseteq \mathbb{Z}^2$ allora valgono le seguenti proprietà:*

(1) *$A \subseteq B$ allora $(A \circ C) \subseteq (B \circ C)$*

(2) *Se $A \subseteq B$ allora $A \circ C \subseteq B \circ C$*

(3) *$A \circ B \subseteq A$*

(4) $(A \circ B) \ominus B = A \ominus B$

(5) $(A \oplus B) \circ B = A \oplus B$

(6) $(A \circ B) \circ B = A \circ B$ *(idempotenza)*

Dimostrazione. (1) Sia $x \in A \circ C$, allora per definzione, $x \in (A \ominus C) \oplus C$. Dunque $x = l + c$ con $l \in A \ominus C$ e $c \in C$. Ma dal fatto che $l \in A \ominus C$, per definizione di erosione, $C_l \subseteq A \subseteq B$ per ipotesi. Allora $l \in B \ominus C$. Dunque, dal momento che $x = l + c$ con $l \in B \ominus C$ e $c \in C$, allora $x \in (B \ominus C) \oplus C = B \circ C$.
(2) Supponiamo che $A \subseteq B$. Per la Proposizione 3.2.7 (2), si ha che $A \ominus C \subseteq B \ominus C$ e da qui, per la Proposizione 3.2.10 (4), ne segue che $(A \ominus C) \oplus C \subseteq (B \ominus C) \oplus C$, che per definizione di apertura equivale a $A \circ C \subseteq B \circ C$.
(3) Sia $x \in A \circ B = (A \ominus B) \oplus B$, allora $x = l + b$ con $l \in A \ominus B$ e $b \in B$. Dal fatto che $l \in A \ominus B$, per definizione di erosione, si ha che $B_l \subseteq A$ con $B_l = \{b + l : b \in B\}$. Quindi in particolare, essendo $x = l + b$, $x \in B_l \subseteq A$. Dunque $x \in A$.
(4) Dalla proprietà 3.2.3 di questa proposizione, sappiamo che $A \circ B \subseteq A$, dunque la prima inclusione è verificata applicando la Proposizione 3.2.7 (2) sulla monotonia dell'erosione, infatti $(A \circ B) \ominus B \subseteq A \ominus B$. Viceversa, per ogni $x \in A \ominus B$, per definizione di erosione, si ha che $B_x \subseteq A$. Ma allora, per ogni $b \in B$ si ha che $b + x \in A$ e dal momento che $x \in A \ominus B$, dal fatto che la dilatazione coincide con la Somma di Minkowski, si ha che $x + b \in (A \circ B) \ominus B$, che equivale a dire che $x + b \in A \circ B$ per definizione di apertura. Pertanto, dal fatto che per ogni $b \in B$ si ha che $x + b \in A \circ B$, ne segue che $B_x \subseteq A \circ B$, che per definizione di erosione, significa che $x \in (A \circ B) \ominus B$. Ciò prova l'inclusione $A \ominus B \subseteq (A \circ B) \ominus B$ e quindi l'uguaglianza.
(5) Per la proprietà 3.2.3 si ha subito che $(A \oplus B) \circ B \subseteq A \oplus B$.
Viceversa, per ogni $x \in A \oplus B$, sapendo che la dilatazione coincide con la somma di Minkowski, si ha che $x \in A + B$ e dunque che esiste qualche $y \in A$ e qualche $z \in B$ tali che $x = y + z$. Essendo $A \oplus B = B \oplus A = \bigcap_{a \in A} B_a$ e valendo la proprietà commutativa della dilatazione, si ha in particolare che $B_y \subseteq A \oplus B$ e dunque, per definizione di erosione, che $y \in (A \oplus B) \ominus B$. Pertanto, essendo $x = y + z = z + y$ con $y \in (A \oplus B) \ominus B$ e $z \in B$, utilizzando nuovamente il fatto che la somma di Minkowski coincide con la dilatazione e per la definizione di apertura, si conclude che $x \in ((A \oplus B) \ominus B) \oplus B = (A \oplus B) \circ B$. Con ciò resta provato che vale anche l'inclusione $A \oplus B \subseteq (A \oplus B) \circ B$ e quindi l'uguaglianza.
(6) Per la definizione di apertura, si ha subito che $(A \circ B) \circ B = ((A \ominus B) \oplus B) \circ B$. Applicando la proprietà 3.2.3 all'insieme $A \ominus B$, si ha che $(A \circ B) \circ B = ((A \ominus B) \oplus B) \circ B = (A \ominus B) \oplus B = A \circ B$ per definizione di apertura.

□

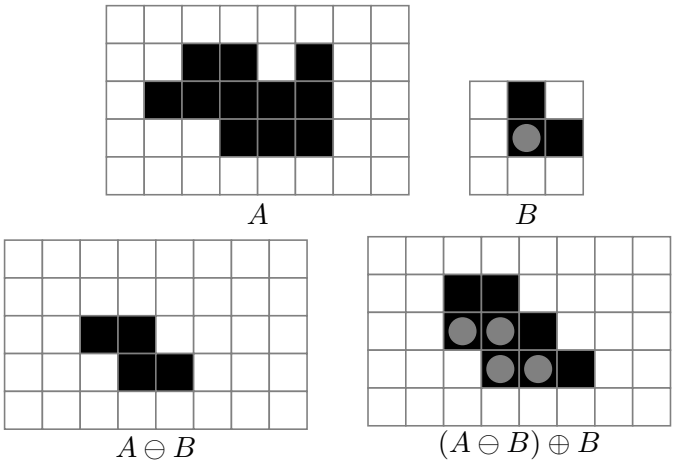


Figura 3.6: Apertura $A \circ B = (A \ominus B) \oplus B$

3.2.4 Chiusura binaria

Definizione 3.2.15. [14] *Siano A, $B \subseteq \mathbb{Z}^2$ due insiemi discreti. Diciamo **chiusura** (più propriamente chiusura binaria) di A rispetto all'elemento strutturante B e lo indichiamo con $A \bullet B$, l'erosione della dilatazione di A rispetto allo stesso elemento strutturante B, cioè*

$$A \bullet B = (A \oplus B) \ominus B$$

Proposizione 3.2.16. [14] *Siano A, $B \subseteq \mathbb{Z}^2$ allora valgono le seguenti proprietà di dualità che legano apertura e chiusura:*

(1) $A \circ B = (A^C \bullet \breve{B})^C$

(2) $A \bullet B = (A^C \circ \breve{B})^C$

Dimostrazione. (1) Per definizione si sa che $(A \bullet B)^{\complement} = ((A \oplus B) \ominus B)^{\complement}$. L'asserto è subito provato applicando a $((A \oplus B) \ominus B)^{\complement}$ la proprietà 3.2.2 della dualità tra erosione e dilatazione. Infatti si ha che:

$$\begin{aligned}(A \bullet B)^{\complement} &= ((A \oplus B) \ominus B)^{\complement} \\ &= (A \oplus B)^{\complement} \oplus \breve{B} \\ &= (A^{\complement} \ominus \breve{B}) \oplus \breve{B}\end{aligned}$$

che per definizione di apertura binaria risulta essere

$$= (A^{\complement} \circ \breve{B})$$

Da qui discende che $A \circ B = (A^C \bullet \breve{B})^C$, ovvero la tesi.
(2) L'asserto discende direttamente dalla proprietà 3.2.4 dell'apertura applicata a $(A^{\complement} \circ \breve{B})$. Infatti, $(A^{\complement} \circ \breve{B}) = \left((A^{\complement})^{\complement} \bullet \left(\breve{\breve{B}}\right)\right)^{\complement} = (A \bullet B)^{\complement}$. Passando al complementare ambo i membri otteniamo la tesi, cioè $A \bullet B = (A^{\complement} \circ \breve{B})^{\complement}$. □

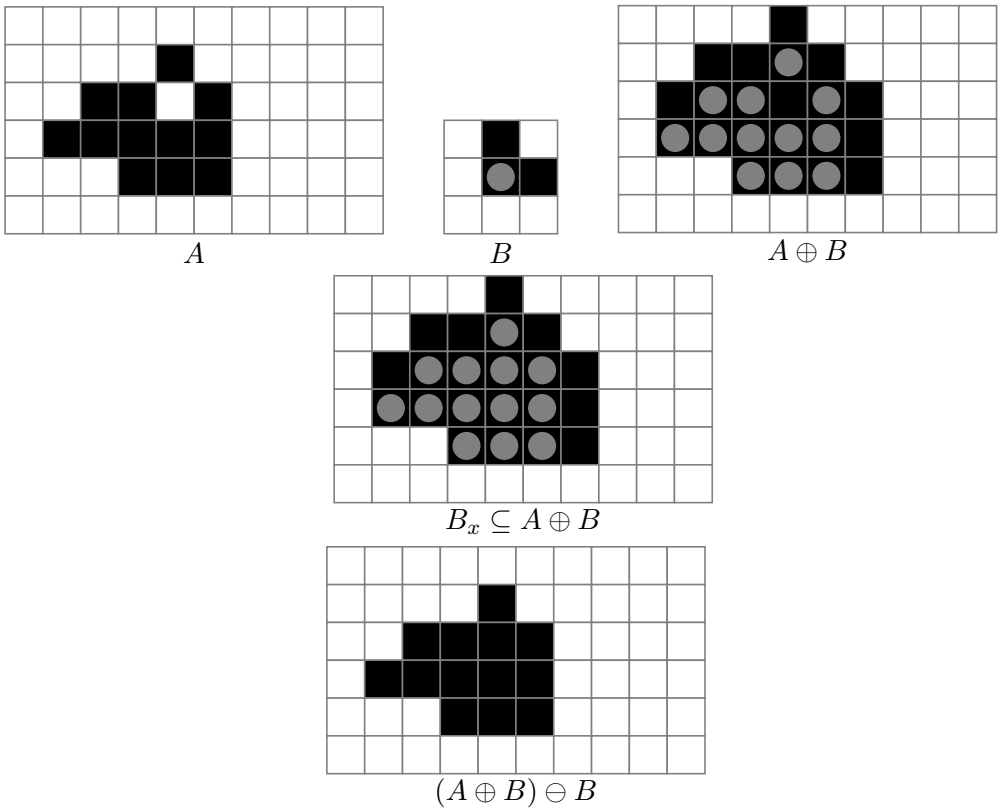


Figura 3.7: Chiusura $A \bullet B = (A \oplus B) \ominus B$

Proposizione 3.2.17. [3, 14] *Siano A, B, C* $\subseteq \mathbb{Z}^2$ *allora valgono le seguenti proprietà:*

(1) $A \subseteq A \bullet B$

(2) *Se* $A \subseteq B$ *allora* $A \bullet C \subseteq B \bullet C$

(3) $(A \bullet B) \oplus B = A \oplus B$

(4) $(A \ominus B) \bullet B = A \ominus B$

(5) $(A \bullet B) \bullet B = A \bullet B$ *(idempotenza)*

Dimostrazione. (1) Infatti, applicando la Proprietà 3.2.3 della Proposizione 3.2.14 agli insiemi $A^{\complement}$ e $\breve{B}$ si ha che $A^{\complement} \circ \breve{B} \subseteq A^{\complement}$ e passando ai complementari (invertendo il verso dell'inclusione) otteniamo $(A^{\complement})^{\complement} \subseteq (A^{\complement} \circ \breve{B})^{\complement}$ ovvero $A \subseteq (A^{\complement} \circ \breve{B})^{\complement}$. Ricordando la Proposizione 3.2.16 di dualità tra gli operatori di chiusura e di apertura, risulta che $A \bullet B = (A^{\complement} \circ \breve{B})^{\complement}$, ne segue che $A \subseteq A \bullet B$.
(2) Supponiamo che $A \subseteq B$. Per la Proposizione 3.2.10 (4) della monotonia della dilatazione, si ha che $A \oplus C \subseteq B \oplus C$ e da qui, per la Proposizione 3.2.7 (2) della monotonia dell'erosione, ne segue che $(A \oplus C) \ominus C \subseteq (B \oplus C) \ominus C$ che, per la definizione di chiusura, equivale a $A \bullet C \subseteq B \bullet C$.
(3) Dalla Proprietà 3.2 di questa Proposizione sappiamo che $A \subseteq A \bullet B$ e per la Proposizione 3.2.10 (4) della monotonia della dilatazione, si ha che $A \oplus B \subseteq (A \bullet B) \oplus B$. Viceversa, per ogni $x \in (A \bullet B) \oplus B$, poiché la dilatazione coincide con la somma di Minkowski, si ha che esiste qualche $y \in A \bullet B$ e qualche $b \in B$ tale che $x = y + b$. Dal fatto che $y \in A \bullet B$, dalla definizione di chiusura, si ha che $y \in (A \oplus B) \ominus B$ e quindi, per definizione di erosione, ne segue che $B_y \subseteq A \oplus B$. Da qui, essendo $b \in B$, per definizione di traslato di un insieme, si ha che $y + b \in B_y$ e quindi $x = y + b \in A \oplus B$. Dunque resta provato che

$(A \bullet B) \oplus B \subseteq A \oplus B$ e quindi l'uguaglianza.
(4) Dalla Proprietà 3.2 di questa Proposizione sappiamo che $A \ominus B \subseteq (A \ominus B) \bullet B$. Viceversa, per ogni $x \in (A \ominus B) \bullet B$, per la definizione di chiusura si ha che $x \in ((A \ominus B) \oplus B) \ominus B$ e da qui, per definizione di erosione, ne segue che $B_x \subseteq (A \ominus B) \oplus B$ che, per definizione di apertura, equivale a dire che $B_x \subseteq A \circ B$. Ma dal fatto che per la Proposizione 3.2.14 3.2.3 sappiamo che $A \circ B \subseteq A$, ne segue che $B_x \subseteq A$ che per definizione di erosione, significa che $x \in A \ominus B$. Ciò prova l'inclusione $(A \ominus B) \bullet B \subseteq A \ominus B$ e quindi l'uguaglianza.
(5) Infatti, per la definizione di chiusura, si ha subito che:

$$(A \bullet B) \bullet B = ((A \bullet B) \oplus B) \ominus B$$

(applicando la Proprietà 3.2 di questa proposizione sulla dilatazione della chiusura)

$$= (A \oplus B) \ominus B$$

(usando ancora la definizione di chiusura)

$$= A \bullet B$$

□

Capitolo 4

Approccio neutrosofico alla Morfologia Matematica

In questo capitolo analizzeremo il concetto di immagine neutrosofica ed introdurremo un nuovo approccio alla Morfologia Matematica che si basa sulla teoria degli insiemi neutrosofici. Saranno definiti gli operatori morfologici neutrosofici di base quali *dilatazione* ed *erosione neutrosofica* ed altri operatori ottenuti dalla composizione di quest'ultimi come *apertura* e *chiusura neutrosofica* rispetto ad un *elemento strutturante* in termini del loro grado di appartenenza, indeterminatezza e non appartenenza.

4.1 Immagini Neutrosofiche

Per poter trattare adeguatamente le immagini digitali in condizioni di incertezza dovute ad esempio ad una scarsa qualità di acquisizione o ad un suo successivo deterioramento che comporta perdita di informazioni, ad esempio per via di una sua compressione, è necessario ricorrere al concetto di immagine digitale neutrosofica (o, brevemente, di immagine neutrosofica) estendendo in maniera appropriata la definizione classica di immagine digitale (a livelli di grigi e binaria) già introdotta nella sezione 3.1. In linea con i principi generali della Teoria Neutrosofica, data una immagine digitale $I : \mathbb{Z}^2 \to T$ con $T = \{0, 1, 2, \dots 2^n - 1\}$ insieme dei valori discreti di intensità (con $n \in \mathbb{N}$), ovvero – nel caso di applicazioni reali – una funzione $I : [x_{min}, x_{max}]_{\mathbb{Z}} \times [y_{min}, y_{max}]_{\mathbb{Z}} \to T$ l'idea fondamentale consiste nell'attribuire al valore di intensità discreto $I(x, y)$ di ciascun pixel di coordinate (x, y) una terna di valori reali $(\mu_I(x, y), \sigma_I(x, y), \omega_I(x, y))$ compresi nell'intervallo unitario che rappresentano rispettivamente il grado di appartenenza (o verità), di indeterminazione e di non appartenenza (o falsità) o – se si vuole – il grado di vicinanza, incertezza o lontananza dal valore indicato per la sua intensità.

In particolare, nel caso di immagini neutrosofiche binarie, in cui i valori di intensità si riducono a 0 (pixel nero o spento) e 1 (pixel bianco o acceso), ad ogni pixel di coordinate (x, y) faremo corrispondere direttamente la terna di gradi di appartenenza $\widetilde{I}(x, y) = (\mu_I(x, y), \sigma_I(x, y), \omega_I(x, y))$ con la convenzione che:

- il valore $(1, 1, 0)$ rappresenterà il pixel neutrosofico completamente acceso, e
- il valore $(0, 0, 1)$ rappresenterà il pixel neutrosofico completamente spento

mentre ogni valore intermedio $\widetilde{I}(x, y) = (\mu_I(x, y), \sigma_I(x, y), \omega_I(x, y))$ rappresenterà un diverso stato di incertezza circa l'accensione del pixel.

Possiamo quindi definire ***immagine digitale neutrosofica*** (o, per brevità, immagine neutrosofica) una funzione $\widetilde{I} : \mathbb{U} \to [0,1]^3$ che ad ogni pixel (x,y) dell'insieme universo $\mathbb{U}$ fa corrispondere la terna $\widetilde{I}(x,y) = (\mu_I(x,y), \sigma_I(x,y), \omega_I(x,y))$ di numeri reali compresi tra 0 ed 1 che esprimono i valori dei gradi di appartenenza (o accensione), indeterminatezza e non appartenenza (o spegnimento) del pixel, dove l'insieme universo è l'intero piano discreto $\mathbb{U} = \mathbb{Z}^2$ o – nel caso concreto di applicazioni computazionali – un suo sottoinsieme finito espresso come prodotto di intervalli di numeri relativi, ossia $\mathbb{U} = [x_{min}, x_{max}]_{\mathbb{Z}} \times [y_{min}, y_{max}]_{\mathbb{Z}}$.

Una ulteriore semplificazione che risulta di grande utilità dal punto di vista applicativo, consiste nel discretizzare l'immagine, ossia nel sostituire (opportunamente scalati) i valori reali e continui della tripla $(\mu_I(x,y), \sigma_I(x,y), \omega_I(x,y))$ con dei valori discreti variabili in un intervallo del tipo $T = \{0, 1, 2, \ldots 2^n - 1\}$ con $n \in \mathbb{N}$; ossia - più formalmente - nel considerare immagini neutrosofiche le funzioni del tipo $\widetilde{I} : \mathbb{U} \to T^3$. Ciò si ottiene semplicemente applicando la trasformazione di scala:

$$\varphi_n : [0,1] \to [0, 2^n - 1]_{\mathbb{Z}} \quad \text{definita da} \quad \varphi_n(t) = \lfloor t \cdot (2^n - 1) \rfloor$$

per un opportuno $n \in \mathbb{N}$ corrispondente al numero dei bit di codifica, simultaneamente a tutte e tre le funzioni di appartenenza $\mu_I : \mathbb{U} \to [0,1]$, indeterminatezza $\sigma_I : \mathbb{U} \to [0,1]$ e non appartenenza $\omega_I : \mathbb{U} \to [0,1]$ dell'immagine neutrosofica $\widetilde{I} = (\mu_I, \sigma_I, \omega_I) : \mathbb{U} \to [0,1]^3$, ossia ponendo:

$$\mu_{I'} = \varphi_n \circ \mu_I : \mathbb{U} \to T \quad , \quad \sigma_{I'} = \varphi_n \circ \sigma_I : \mathbb{U} \to T \quad \text{e} \quad \omega_{I'} = \varphi_n \circ \omega_I : \mathbb{U} \to T$$

ottenendo quindi la discretizzazione $\widetilde{I'} = (\mu_{I'}, \sigma_{I'}, \omega_{I'}) : \mathbb{U} \to T^3$ dell'immagine neutrosofica $\widetilde{I}$.

Nel caso particolare in cui si pone $\mathbb{U} = \{0,1\}$ ed $n = 8$, ossia $T = \{0, \ldots 255\}$, ciò consente di rappresentare graficamente (non di descrivere matematicamente) una immagine neutrosofica $\widetilde{I} : \mathbb{U} \to T^3$ come una tripla $(\mu_I, \sigma_I, \omega_I)$, di immagini a livelli di grigio $\mu_I : \mathbb{U} \to T$, $\sigma_I : \mathbb{U} \to T$ e $\omega_I : \mathbb{U} \to T$ tutte definite sullo stesso dominio $\mathbb{U}$ dell'immagine originaria (ed aventi quindi le sue stesse dimensioni $w_I = x_{max} - x_{min} + 1$ ed $h_I = y_{max} - y_{min} + 1$ nel caso di sottoinsieme discreto del tipo $[x_{min}, x_{max}]_{\mathbb{Z}} \times [y_{min}, y_{max}]_{\mathbb{Z}}$) e rappresentanti rispettivamente i gradi di accensione, indeterminatezza e spegnimento dei relativi pixel (si veda la Figura 4.1).

D'ora in poi, per tutti gli insiemi corrispondenti ad immagini digitali neutrosofiche considereremo come insieme universo $\mathbb{U}$ il piano discreto $\mathbb{Z}^2$ o – nel caso concreto di applicazioni computazionali – un suo sottoinsieme finito espresso come prodotto di intervalli di numeri relativi, ossia $[x_{min}, x_{max}]_{\mathbb{Z}} \times [y_{min}, y_{max}]_{\mathbb{Z}}$.

Il modo in cui ad una immagine digitale acquisita $I : \mathbb{U} \to T$ si associa una immagine neutrosofica $\widetilde{I} = (\mu_I, \sigma_I, \omega_I) : \mathbb{U} \to [0,1]^3$, sebbene non univoco, è comunque ampiamente accettato in letteratura [2, 5, 6, 15, 20] e si basa sull'idea di valutare localmente la distribuzione delle intensità (ad esempio dei livelli di grigio) di ogni singolo pixel in un suo opportuno intorno.

Più precisamente, data una immagine digitale $I : \mathbb{U} \to [0,1]$, sia essa a livelli di grigio oppure binaria, per ottenere la corrispondente immagine neutrosofica $\widetilde{I} = (\mu_I, \sigma_I, \omega_I) : \mathbb{U} \to [0,1]^3$, per ogni pixel di coordinate $(x,y) \in \mathbb{U}$ si calcolano i gradi di appartenenza

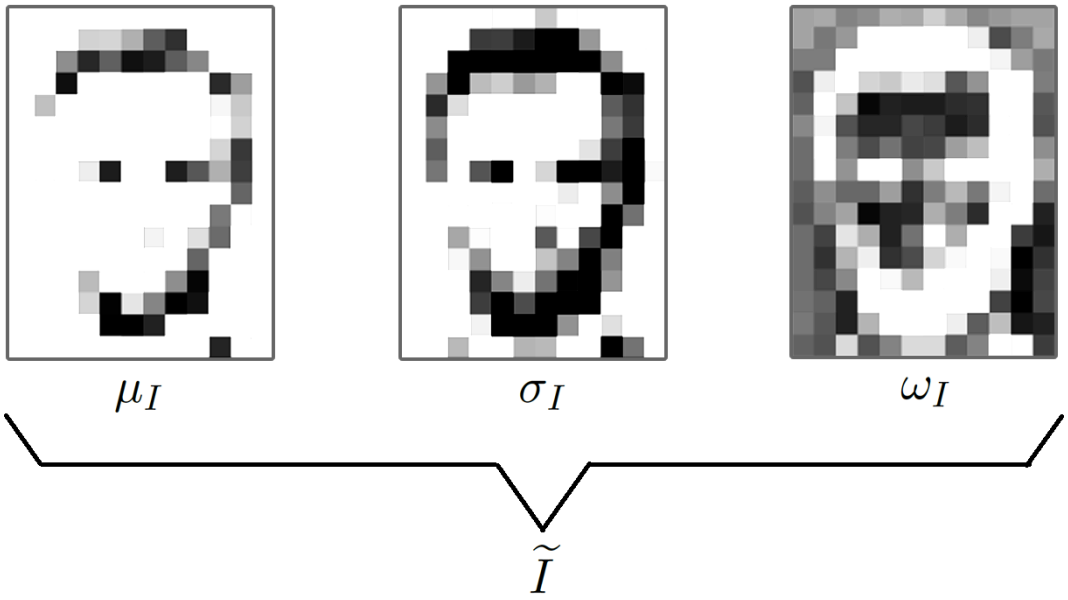


Figura 4.1: Esempio di rappresentazione di una immagine neutrosofica $\widetilde{I}$ nelle sue tre componenti μ_I, σ_I e ω_I.

$\mu_I(x,y)$, indeterminatezza $\sigma_I(x,y)$ e non appartenenza $\omega_I(x,y)$, ossia:

$$\mu_I(x,y) = \frac{\overline{I}(x,y) - \overline{I}_{min}}{\overline{I}_{max} - \overline{I}_{min}}$$

$$\sigma_I(x,y) = \frac{\delta(x,y) - \delta_{min}}{\delta_{max} - \delta_{max}}$$

$$\omega_I(x,y) = 1 - \mu_I(x,y) = \frac{\overline{I}_{max} - \overline{I}(x,y)}{\overline{I}_{max} - \overline{I}_{min}}$$

dove:

- la funzione $\overline{I}$ valuta la media delle intensità dei pixel dell'immagine I in un intorno quadrato di lato $\omega + 1$ prefissato, con ω intero positivo pari ed avente il centro nel generico pixel di coordinate $(x,y) \in \mathbb{U}$ ed è definita secondo la formula:

$$\overline{I}(x,y) = \frac{1}{(\omega+1)^2} \sum_{i=x-\frac{\omega}{2}}^{x+\frac{\omega}{2}} \sum_{j=y-\frac{\omega}{2}}^{y+\frac{\omega}{2}} I(i,j) \tag{4.1}$$

- i valori di picco massimo e minimo delle intensità media $\overline{I}$ sono ovviamente definiti da:

$$\overline{I}_{max} = \max_{(x,y)\in\mathbb{U}} \overline{I}(x,y) \quad \text{e} \quad \overline{I}_{min} = \min_{(x,y)\in\mathbb{U}} \overline{I}(x,y) \tag{4.2}$$

- la funzione di omogeneità δ fornisce una misura dello scostamento tra l'intensità di ciascun pixel dell'immagine I e la sua intensità media $\overline{I}$ nell'intorno considerato e, per ogni $(x,y) \in \mathbb{U}$, è definita espressamente da:

$$\delta(x,y) = \left| I(x,y) - \overline{I}(x,y) \right| \tag{4.3}$$

- i valori di picco massimo e minimo della funzione di omogeneità δ sono dati da:

$$\delta_{max} = \max_{(x,y)\in\mathbb{U}} \delta(x,y) \quad \text{e} \quad \delta_{min} = \min_{(x,y)\in\mathbb{U}} \delta(x,y) \tag{4.4}$$

Il processo di trasformazione da immagine digitale classica ad immagine digitale neutrosofica appena descritto è schematicamente rappresentato nella Figura 4.2).

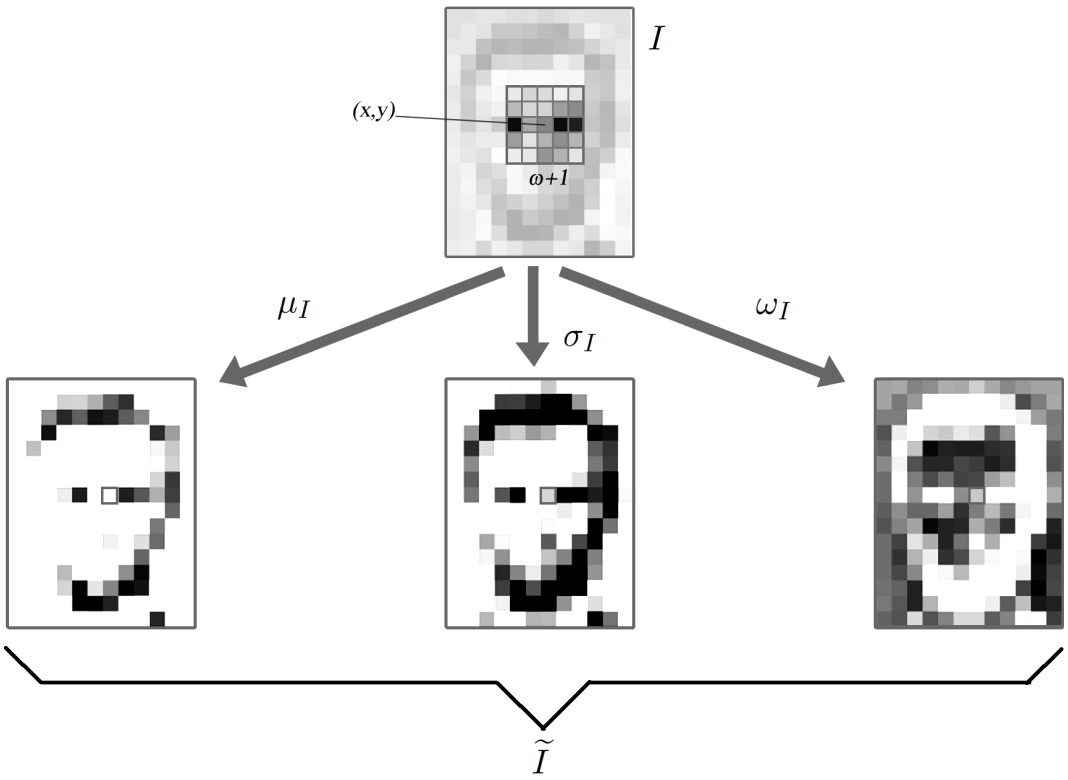


Figura 4.2: Conversione di una immagine digitale I nella sua corrispondente immagine neutrosofica $\widetilde{I}$.

Come è facile desumere e verificare, la scelta del lato (o, se si vuole, diametro) ω dell'intorno condiziona fortemente, a cascata, il calcolo dei valori della funzione di intensità media $\overline{I}$, della funzione di omogeneità δ ed infine dei gradi di appartenenza $\mu_I(x,y)$, indeterminatezza $\sigma_I(x,y)$ e non appartenenza $\omega_I(x,y)$ di ogni singolo pixel dell'immagine. In particolare, valori piccoli come $\omega = 2$ (che corrisponde alla scelta di un intorno di lato 3) oppure $\omega = 4$ (che corrisponde ad un intorno di lato 5) forniscono una minore approssimazione delle intensità dei pixel contigui mentre valori più alti rischiano di compromettere (rallentando con complessità quadratica) i tempi di calcolo in caso di applicazioni computazionali.

Analogamente a quanto visto nel caso delle immagini digitali binarie, in cui quest'ultime potevano essere identificate con un sottoinsieme del piano discreto $\mathbb{Z}^2$, anche le immagini digitali neutrosofiche possono essere trattate più semplicemente come sottoinsiemi neutrosofici dell'insieme universo $\mathbb{U} = \mathbb{Z}^2$.

4.2 Morfologia Neutrosofica

Una volta che – come visto nella sezione precedente – a partire da una immagine digitale I binaria o a livelli di grigi, si ottiene la corrispondente immagine neutrosofica $\widetilde{I}$, è auspicabile che su quest'ultima possano essere effettuate operazioni di filtraggio atte ad isolare e rimuovere parti indesiderate o ad evidenziarne alcune caratteristiche, in maniera analoga a quanto è possibile fare con gli operatori morfologici binari che abbiamo presentato nel Capitolo 3. Questo è esattamente quello che faremo nel seguito, generalizzando opportunamente sui domini neutrosofici i ben noti operatori morfologici di dilatazione, erosione, apertura e chiusura ed ogni altra nozione collegata e necessaria.

Definizione 4.2.1. [11] *Sia $\widetilde{B} = \langle \mathbb{U}, \mu_B, \sigma_B, \omega_B \rangle$ un SVN-set sull'insieme universo $\mathbb{U} = \mathbb{Z}^2$. Diciamo **riflessione** di $\widetilde{B}$ rispetto all'origine e lo indichiamo con $-\widetilde{B}$ l'insieme:*

$$-\widetilde{B} = \langle \mathbb{U}, -\mu_B, -\sigma_B, -\omega_B \rangle$$

dove le funzioni di appartenenza, indeterminatezza e non appartenenza sono definite rispettivamente da $-\mu_B(v) = \mu_B(-v)$, $-\sigma_B(v) = \sigma_B(-v)$ e $-\omega_B(v) = \omega_B(-v)$ per ogni $v \in \mathbb{U}$.

Definizione 4.2.2. [11] *Sia $\widetilde{A} = \langle \mathbb{U}, \mu_A, \sigma_A, \omega_A \rangle$ un SVN-set sull'insieme universo $\mathbb{U}$ e sia $p \in \mathbb{U}$. Diciamo **traslato** di $\widetilde{A}$ tramite p e lo indichiamo con $\widetilde{A_p}$ l'insieme:*

$$\widetilde{A_p} = \left\langle \mathbb{U}, \mu_{A_p}, \sigma_{A_p}, \omega_{A_p} \right\rangle$$

dove le funzioni di appartenenza, indeterminatezza e non appartenenza sono definite rispettivamente da $\mu_{A_p}(v) = \mu_A(v+p)$, $\sigma_{A_p}(v) = \sigma_A(v+p)$ e $\omega_{A_p}(v) = \omega_A(v+p)$ per ogni $v \in \mathbb{U}$.

Le operazioni come *erosione, dilatazione, apertura* e *chiusura neutrosofica* di un'immagine neutrosofica tramite l'elemento strutturante neutrosofico, sono definite in termini delle funzioni di appartenenza, indeterminatezza e non appartenenza.

4.2.1 Dilatazione Neutrosofica

Analogamente a quanto già visto nel caso delle immagini binarie, l'operatore di dilatazione neutrosofica utilizza una seconda immagine quale elemento strutturante.

Definizione 4.2.3. [12] *Siano $\widetilde{A} = \langle \mathbb{U}, \mu_A, \sigma_A, \omega_A \rangle$ e $\widetilde{B} = \langle \mathbb{U}, \mu_B, \sigma_B, \omega_B \rangle$ due SVN-set su uno stesso insieme universo $\mathbb{U}$. Definiamo la **dilatazione neutrosofica** dell'insieme neutrosofico $\widetilde{A}$ rispetto all'elemento strutturante $\widetilde{B}$ come:*

$$\widetilde{A} \widetilde{\oplus} \widetilde{B} = \langle \mathbb{U}, \mu_{A \oplus B}, \sigma_{A \oplus B}, \omega_{A \oplus B} \rangle$$

in cui le funzioni di appartenenza, indeterminatezza e non appartenenza sono definite rispettivamente per ogni $v \in \mathbb{U}$ da:

$$\begin{aligned}
\mu_{A \oplus B}(v) &= \sup_{u \in \mathbb{U}} \min \{\mu_A(v+u), \mu_B(u)\} \\
\sigma_{A \oplus B}(v) &= \sup_{u \in \mathbb{U}} \min \{\sigma_A(v+u), \sigma_B(u)\} \\
\omega_{A \oplus B}(v) &= \inf_{u \in \mathbb{U}} \max \{\omega_A(v+u), 1 - \omega_B(u)\}
\end{aligned}$$

Lemma 4.2.4. [11] *Per ogni* $\widetilde{A} = \langle \mathbb{U}, \mu_A, \sigma_A, \omega_A \rangle \in \mathcal{SVN}(\mathbb{U})$ *e detto* $\widetilde{\mathbb{U}}$ *l'insieme neutrosofico assoluto, si ha che* $\widetilde{A} \widetilde{\oplus} \widetilde{\mathbb{U}} \subseteq \widetilde{A}$.

Dimostrazione. Per definizione, si ha che $\widetilde{A} \widetilde{\oplus} \widetilde{\mathbb{U}} = \langle \mathbb{U}, \mu_{A \oplus \mathbb{U}}, \sigma_{A \oplus \mathbb{U}}, \omega_{A \oplus \mathbb{U}} \rangle$ dove per ogni $v \in \mathbb{U}$ si ha che:

$$\begin{aligned} \mu_{A \oplus \mathbb{U}}(v) &= \sup_{u \in \mathbb{U}} \min \{\mu_A(v+u), 1\} \\ &= \sup_{u \in \mathbb{U}} \mu_A(v+u) \\ &= \mu_A(v) \end{aligned}$$

$$\begin{aligned} \sigma_{A \oplus \mathbb{U}}(v) &= \sup_{u \in \mathbb{U}} \min \{\sigma_A(v+u), 1\} \\ &= \sup_{u \in \mathbb{U}} \sigma_A(v+u) \\ &= \sigma_A(v) \end{aligned}$$

$$\begin{aligned} \omega_{A \oplus \mathbb{U}}(v) &= \inf_{u \in \mathbb{U}} \max \{\omega_A(v+u), 1-0\} \\ &= 1 \\ &= \underline{1}(v) \end{aligned}$$

Poiché evidentemente $\underline{1}(v) \geq \omega_A(v)$ per ogni $v \in \mathbb{U}$, allora vale $\widetilde{A} \widetilde{\oplus} \widetilde{\mathbb{U}} = \langle \mathbb{U}, \mu_A, \sigma_A, \underline{1} \rangle \subseteq \langle \mathbb{U}, \mu_A, \sigma_A, \omega_A \rangle = \widetilde{A}$. □

Lemma 4.2.5. [11] *Per ogni* $\widetilde{A} = \langle \mathbb{U}, \mu_A, \sigma_A, \omega_A \rangle \in \mathcal{SVN}(\mathbb{U})$ *e detto* $\widetilde{\emptyset}$ *l'insieme neutrosofico vuoto, si ha che* $\widetilde{A} \widetilde{\oplus} \widetilde{\emptyset} \subseteq \widetilde{A}^{\complement}$.

Dimostrazione. Per definizione, si ha che $\widetilde{A} \widetilde{\oplus} \widetilde{\emptyset} = \langle \mathbb{U}, \mu_{A \oplus \emptyset}, \sigma_{A \oplus \emptyset}, \omega_{A \oplus \emptyset} \rangle$ dove per ogni $v \in \mathbb{U}$ si ha che:

$$\begin{aligned} \mu_{A \oplus \emptyset}(v) &= \sup_{u \in \mathbb{U}} \min \{\mu_A(v+u), 0\} \\ &= 0 \\ &= \underline{0}(v) \end{aligned}$$

$$\begin{aligned} \sigma_{A \oplus \emptyset}(v) &= \sup_{u \in \mathbb{U}} \min \{\sigma_A(v+u), 0\} \\ &= 0 \\ &= \underline{0}(v) \end{aligned}$$

$$\begin{aligned} \omega_{A \oplus \emptyset}(v) &= \inf_{u \in \mathbb{U}} \max \{\omega_A(v+u), 1-1\} \\ &= \inf_{u \in \mathbb{U}} \omega_A(v+u) \\ &= \omega_A(v) \end{aligned}$$

Dal momento che $\mu_{A^{\complement}} = 1 - \mu_A$, $\sigma_{A^{\complement}} = 1 - \sigma_A$ e $\omega_{A^{\complement}} = 1 - \omega_A$, si può affermare che:

$$\begin{aligned} \mu_{A \oplus \mathbb{U}} &= \underline{0} \leq 1 - \mu_A \\ \sigma_{A \oplus \mathbb{U}} &= \underline{0} \leq 1 - \sigma_A \\ \omega_{A \oplus \mathbb{U}} &= \omega_A \geq 1 - \omega_A \end{aligned}$$

Dunque, $\widetilde{A} \widetilde{\oplus} \widetilde{\mathbb{U}} \subseteq \widetilde{A}^{\complement}$. □

Proposizione 4.2.6. [11] *Siano* $\widetilde{A} = \langle \mathbb{U}, \mu_A, \sigma_A, \omega_A \rangle$, $\widetilde{B} = \langle \mathbb{U}, \mu_B, \sigma_B, \omega_B \rangle$, $\widetilde{C} = \langle \mathbb{U}, \mu_C, \sigma_C, \omega_C \rangle$ *SVN-set in* $\mathcal{SVN}(\mathbb{U})$ *tali che* $\widetilde{A} \Subset \widetilde{B}$. *Allora* $\widetilde{A} \widetilde{\oplus} \widetilde{C} \Subset \widetilde{B} \widetilde{\oplus} \widetilde{C}$.

Dimostrazione. Essendo $\widetilde{A} \Subset \widetilde{B}$, per definizione di sottoinsieme neutrosofico, si ha che $\mu_A(v) \leq \mu_B(v)$, $\sigma_A(v) \leq \sigma_B(v)$, $\omega_A(v) \geq \omega_B(v)$ per ogni $v \in \mathbb{U}$. Per ogni $v \in \mathbb{U}$, si ha che:

$$\begin{aligned} \mu_{A \oplus C}(v) &= \sup_{u \in \mathbb{U}} \min \{\mu_A(v+u), \mu_C(u)\} \\ &\leq \sup_{u \in \mathbb{U}} \min \{\mu_B(v+u), \mu_C(u)\} \\ &= \mu_{B \oplus C}(v) \end{aligned}$$

$$\begin{aligned} \sigma_{A \oplus C}(v) &= \sup_{u \in \mathbb{U}} \min \{\sigma_A(v+u), \sigma_C(u)\} \\ &\leq \sup_{u \in \mathbb{U}} \min \{\sigma_B(v+u), \sigma_C(u)\} \\ &= \sigma_{B \oplus C}(v) \end{aligned}$$

$$\begin{aligned} \omega_{A \oplus C}(v) &= \inf_{u \in \mathbb{U}} \max \{\omega_A(v+u), 1 - \omega_C(u)\} \\ &\geq \inf_{u \in \mathbb{U}} \max \{\omega_B(v+u), 1 - \omega_C(u)\} \\ &= \omega_{B \oplus C}(v) \end{aligned}$$

Quindi $\widetilde{A} \widetilde{\oplus} \widetilde{C} \Subset \widetilde{B} \widetilde{\oplus} \widetilde{C}$.

□

Proposizione 4.2.7. [11] [12] *Siano* $\left\{ \widetilde{A}_i \right\}_{i \in I}$ *una famiglia di SVN-set* $\widetilde{A}_i = \langle \mathbb{U}, \mu_{A_i}, \sigma_{A_i}, \omega_{A_i} \rangle$ *e* $\widetilde{B} = \langle \mathbb{U}, \mu_B, \sigma_B, \omega_B \rangle$ *un SVN-set su uno stesso insieme universo* $\mathbb{U}$, *allora si ha che:*

(1) $\left(\bigcap_{i \in I} \widetilde{A}_i \right) \widetilde{\oplus} \widetilde{B} \Subset \bigcap_{i \in I} (\widetilde{A}_i \widetilde{\oplus} \widetilde{B})$

(2) $\bigcup_{i \in I} (\widetilde{A}_i \widetilde{\oplus} \widetilde{B}) \Subset \left(\bigcup_{i \in I} \widetilde{A}_i \right) \widetilde{\oplus} \widetilde{B}$

Dimostrazione. (1) Per ogni $v \in \mathbb{U}$, si ha che:

$$\begin{aligned} \mu_{(\mathbb{m}_{i \in I} A_i) \oplus B}(v) &= \sup_{u \in \mathbb{U}} \min \left\{ \mu_{(\mathbb{m}_{i \in I} A_i)}(v+u), \mu_B(u) \right\} \\ &= \sup_{u \in \mathbb{U}} \min \left\{ \inf_{i \in I} \mu_{A_i}(v+u), \mu_B \right\} \\ &= \sup_{u \in \mathbb{U}} \inf_{i \in I} \{ \min \mu_{A_i}(v+u), \mu_B \} \\ &\leq \inf_{i \in I} \sup_{u \in \mathbb{U}} \{ \min \mu_{A_i}(v+u), \mu_B(u) \} \\ &= \bigwedge_{i \in I} \sup_{u \in \mathbb{U}} \min \{ \mu_{A_i}(v+u), \mu_B(u) \} \\ &= \bigwedge_{i \in I} \mu_{A_i \oplus B}(v) \\ &= \mu_{\mathbb{m}_{i \in I}(A_i \oplus B)}(v) \end{aligned}$$

$$\begin{aligned}
\sigma_{(\mathbb{m}_{i\in I}A_i)\oplus B}(v) &= \sup_{u\in\mathbb{U}} \min\left\{\sigma_{(\mathbb{m}_{i\in I}A_i)}(v+u), \sigma_B(u)\right\} \\
&= \sup_{u\in\mathbb{U}} \min\left\{\inf_{i\in I}\sigma_{A_i}(v+u), \sigma_B\right\} \\
&= \sup_{u\in\mathbb{U}} \inf_{i\in I}\left\{\min \sigma_{A_i}(v+u), \sigma_B\right\} \\
&\leq \inf_{i\in I} \sup_{u\in\mathbb{U}}\left\{\min \sigma_{A_i}(v+u), \sigma_B(u)\right\} \\
&= \bigwedge_{i\in I} \sup_{u\in\mathbb{U}} \min\left\{\sigma_{A_i}(v+u), \sigma_B(u)\right\} \\
&= \bigwedge_{i\in I} \sigma_{A_i\oplus B}(v) \\
&= \sigma_{\mathbb{m}_{i\in I}(A_i\oplus B)}(v)
\end{aligned}$$

$$\begin{aligned}
\omega_{(\mathbb{m}_{i\in I}A_i)\oplus B}(v) &= \inf_{u\in\mathbb{U}} \max\left\{\omega_{(\mathbb{m}_{i\in I}A_i)}(v+u), 1-\omega_B(u)\right\} \\
&= \inf_{u\in\mathbb{U}} \max\left\{\sup_{i\in I}\omega_{A_i}(v+u), 1-\omega_B(u)\right\} \\
&= \inf_{u\in\mathbb{U}} \sup_{i\in I}\left\{\max \omega_{A_i}(v+u), 1-\omega_B(u)\right\} \\
&\geq \sup_{i\in I} \inf_{u\in\mathbb{U}}\left\{\max \omega_{A_i}(v+u), 1-\omega_B(u)\right\} \\
&= \bigvee_{i\in I} \inf_{u\in\mathbb{U}} \max\left\{\omega_{A_i}(v+u), 1-\omega_B(u)\right\} \\
&= \bigvee_{i\in I} \omega_{(A_i\oplus B)}(v) \\
&= \omega_{\mathbb{m}_{i\in I}(A_i\oplus B)}(v)
\end{aligned}$$

Dunque, essendo

$$\begin{aligned}
\mu_{(\mathbb{m}_{i\in I}A_i)\oplus B}(v) &\leq \mu_{\mathbb{m}_{i\in I}(A_i\oplus B)}(v), \\
\sigma_{(\mathbb{m}_{i\in I}A_i)\oplus B}(v) &\leq \sigma_{\mathbb{m}_{i\in I}(A_i\oplus B)}(v)
\end{aligned}$$

e

$$\omega_{(\mathbb{m}_{i\in I}A_i)\oplus B}(v) \geq \omega_{\mathbb{m}_{i\in I}(A_i\oplus B)}(v)$$

per ogni $v \in \mathbb{U}$, resta provata la tesi, cioè che $\left(\bigcap_{i\in I}\widetilde{A}_i\right)\widetilde{\oplus}\widetilde{B} \Subset \bigcap_{i\in I}(\widetilde{A}_i\widetilde{\oplus}\widetilde{B})$.
(2) Per ogni $v \in \mathbb{U}$, si ha che:

$$\begin{aligned}
\mu_{\mathbb{u}_{i\in I}(A_i\oplus B)}(v) &= \sup_{i\in I} \mu_{A\oplus B}(v) \\
&= \sup_{i\in I}\sup_{u\in\mathbb{U}} \min\left\{\mu_{A_i}(v+u), \mu_B(u)\right\} \\
&= \sup_{u\in\mathbb{U}}\sup_{i\in I} \min\left\{\mu_{A_i}(v+u), \mu_B(u)\right\} \\
&\leq \sup_{u\in\mathbb{U}} \min\left\{\mu_{(\mathbb{u}_{i\in I}A_i)}(v+u), \mu_B(u)\right\} \\
&= \mu_{(\mathbb{u}_{i\in I}A_i)\oplus B}(v)
\end{aligned}$$

$$\begin{aligned}
\sigma_{\Cup_{i\in I}(A_i\oplus B)}(v) &= \sup_{i\in I}\sigma_{A\oplus B}(v)\\
&= \sup_{i\in I}\sup_{u\in\mathbb{U}}\min\{\sigma_{A_i}(v+u),\sigma_B(u)\}\\
&= \sup_{u\in\mathbb{U}}\sup_{i\in I}\min\{\sigma_{A_i}(v+u),\sigma_B(u)\}\\
&\leq \sup_{u\in\mathbb{U}}\min\left\{\sigma_{(\Cup_{i\in I}A_i)}(v+u),\sigma_B(u)\right\}\\
&= \sigma_{(\Cup_{i\in I}A_i)\oplus B}(v)
\end{aligned}$$

$$\begin{aligned}
\omega_{\Cup_{i\in I}(A_i\oplus B)}(v) &= \inf_{i\in I}\omega_{A\oplus B}(v)\\
&= \inf_{i\in I}\inf_{u\in\mathbb{U}}\max\{\omega_{A_i}(v+u),\omega_B(u)\}\\
&= \inf_{u\in\mathbb{U}}\inf_{i\in I}\max\{\omega_{A_i}(v+u),\omega_B(u)\}\\
&\geq \inf_{u\in\mathbb{U}}\max\left\{\omega_{(\Cup_{i\in I}A_i)}(v+u),\omega_B(u)\right\}\\
&= \omega_{(\Cup_{i\in I}A_i)\oplus B}(v)
\end{aligned}$$

Quindi, dal momento che

$$\mu_{\Cup_{i\in I}(A_i\oplus B)}(v) \leq \mu_{(\Cup_{i\in I}A_i)\oplus B}(v),$$

$$\sigma_{\Cup_{i\in I}(A_i\oplus B)}(v) \leq \sigma_{(\Cup_{i\in I}A_i)\oplus B}(v)$$

e

$$\omega_{\Cup_{i\in I}(A_i\oplus B)}(v) \geq \omega_{(\Cup_{i\in I}A_i)\oplus B}(v)$$

resta provata la tesi, cioè che $\bigcup_{i\in I}(\widetilde{A}_i\widetilde{\oplus}\widetilde{B}) \Subset \left(\bigcup_{i\in I}\widetilde{A}_i\right)\widetilde{\oplus}\widetilde{B}$. □

4.2.2 Erosione Neutrosofica

Anche per quanto riguarda l'operatore morfologico di erosione è possibile introdurre una generalizzazione in ambito neutrosofico.

Definizione 4.2.8. [12] *Siano $\widetilde{A} = \langle \mathbb{U}, \mu_A, \sigma_A, \omega_A\rangle$ e $\widetilde{B} = \langle \mathbb{U}, \mu_B, \sigma_B, \omega_B\rangle$ due SVN-set su uno stesso insieme universo $\mathbb{U}$. Definiamo l'**erosione neutrosofica** dell'insieme neutrosofico $\widetilde{A}$ rispetto all'elemento strutturante $\widetilde{B}$ come:*

$$A\widetilde{\ominus}B = \langle \mathbb{U}, \mu_{A\ominus B}, \sigma_{A\ominus B}, \omega_{A\ominus B}\rangle$$

dove le funzioni di appartenenza, indeterminatezza e non appartenenza sono definite rispettivamente per ogni $v \in \mathbb{U}$ da:

$$\begin{aligned}
\mu_{A\ominus B}(v) &= \inf_{u\in\mathbb{U}}\max\{\mu_A(v+u), 1-\mu_B(u)\}\\
\sigma_{A\ominus B}(v) &= \inf_{u\in\mathbb{U}}\max\{\sigma_A(v+u), 1-\sigma_B(u)\}\\
\omega_{A\ominus B}(v) &= \sup_{u\in\mathbb{U}}\min\{\omega_A(v+u), \omega_B(u)\}
\end{aligned}$$

Proposizione 4.2.9. [12] *Siano $\widetilde{A} = \langle \mathbb{U}, \mu_A, \sigma_A, \omega_A\rangle$ e $\widetilde{B} = \langle \mathbb{U}, \mu_B, \sigma_B, \omega_B\rangle$ due SVN-set su uno stesso insieme universo $\mathbb{U} = \mathbb{Z}^2$, allora l'erosione e la dilatazione neutrosofica sono duali, cioè:*

$$(\widetilde{A}^{\complement}\widetilde{\oplus}\ \widetilde{B})^{\complement} = \widetilde{A}\widetilde{\ominus}\widetilde{B}$$

Dimostrazione. Per definizione, $(\widetilde{A}^{\complement} \widetilde{\oplus} \widetilde{B})^{\complement} = \left\langle \mathbb{U}, \mu_{(A^{\complement}\oplus B)^{\complement}}, \sigma_{(A^{\complement}\oplus B)^{\complement}}, \omega_{(A^{\complement}\oplus B)^{\complement}} \right\rangle$, dove per ogni $v \in \mathbb{U}$ si ha che:

$$\begin{aligned}
\mu_{(A^{\complement}\oplus B)^{\complement}}(v) &= 1 - \mu_{(A^{\complement}\oplus B)}(v) \\
&= 1 - \sup_{u\in\mathbb{U}} \min\left\{\mu_{A^{\complement}}(v+u), \mu_B(u)\right\} \\
&= \inf_{u\in\mathbb{U}} \left\{1 - \min[\mu_{A^{\complement}}(v+u), \mu_B(u)]\right\} \\
&= \inf_{u\in\mathbb{U}} \max\left\{1 - \mu_{A^{\complement}}(v+u), 1 - \mu_B(u)\right\} \\
&= \inf_{u\in\mathbb{U}} \max\left\{\mu_A(v+u), 1 - \mu_B(u)\right\} \\
&= \sigma_{A\ominus B}(v)
\end{aligned}$$

$$\begin{aligned}
\sigma_{(A^{\complement}\oplus B)^{\complement}}(v) &= 1 - \sigma_{(A^{\complement}\oplus B)}(v) \\
&= 1 - \sup_{u\in\mathbb{U}} \min\left\{\sigma_{A^{\complement}}(v+u), \sigma_B(u)\right\} \\
&= \inf_{u\in\mathbb{U}} \left\{1 - \min[\sigma_{A^{\complement}}(v+u), \sigma_B(u)]\right\} \\
&= \inf_{u\in\mathbb{U}} \max\left\{1 - \sigma_{A^{\complement}}(v+u), 1 - \sigma_B(u)\right\} \\
&= \inf_{u\in\mathbb{U}} \max\left\{\sigma_A(v+u), 1 - \sigma_B(u)\right\} \\
&= \sigma_{A\ominus B}(v)
\end{aligned}$$

$$\begin{aligned}
\omega_{(A^{\complement}\oplus B)^{\complement}}(v) &= 1 - \omega_{(A^{\complement}\oplus B)}(v) \\
&= 1 - \inf_{u\in\mathbb{U}} \max\left\{\omega_{A^{\complement}}(v+u), 1 - \omega_B(u)\right\} \\
&= \sup_{u\in\mathbb{U}} \left\{1 - \max[\omega_{A^{\complement}}(v+u), 1 - \omega_B(u)]\right\} \\
&= \sup_{u\in\mathbb{U}} \min\left\{1 - \omega_{A^{\complement}}(v+u), \omega_B(u)\right\} \\
&= \sup_{u\in\mathbb{U}} \min\left\{\omega_A(v+u), \sigma_B(u)\right\} \\
&= \omega_{A\ominus B}(v)
\end{aligned}$$

Dunque $\left\langle \mathbb{U}, \mu_{(A^{\complement}\oplus B)^{\complement}}, \sigma_{(A^{\complement}\oplus B)^{\complement}}, \omega_{(A^{\complement}\oplus B)^{\complement}} \right\rangle = \langle \mathbb{U}, \mu_{A\ominus B}, \sigma_{A\ominus B}, \omega_{A\ominus B} \rangle$.
Quindi la tesi è provata. □

Proposizione 4.2.10. [11] *Siano* $\widetilde{A} = \langle \mathbb{U}, \mu_A, \sigma_A, \omega_A \rangle$, $\widetilde{B} = \langle \mathbb{U}, \mu_B, \sigma_B, \omega_B \rangle$, $\widetilde{C} = \langle \mathbb{U}, \mu_C, \sigma_C, \omega_C \rangle$ *SVN-set in* $\mathcal{SVN}(\mathbb{U})$*, allora valgono le seguenti proprietà di monotonia:*

(1) $\widetilde{A} \Subset \widetilde{B} \Rightarrow \widetilde{A}\widetilde{\ominus}\widetilde{C} \Subset \widetilde{B}\widetilde{\ominus}\widetilde{C}$

(2) $\widetilde{A} \Subset \widetilde{B} \Rightarrow \widetilde{C}\widetilde{\ominus}\widetilde{B} \Subset \widetilde{C}\widetilde{\ominus}\widetilde{A}$

Dimostrazione. (1) Dall'ipotesi $\widetilde{A} \Subset \widetilde{B}$, per definizione, si ha che $\mu_A(u) \leq \mu_B(u)$, $\sigma_A(u) \leq \sigma_B(u)$, $\omega_A(u) \geq \omega_B(u)$. Per ogni $v \in \mathbb{U}$, si ha:

$$\begin{aligned}
\mu_{A\ominus C}(v) &= \inf_{u\in\mathbb{U}} \max\left\{\mu_A(v+u), 1 - \mu_C(u)\right\} \\
&\leq \inf_{u\in\mathbb{U}} \max\left\{\mu_B(v+u), 1 - \mu_C(u)\right\} \\
&= \mu_{B\ominus C}(v)
\end{aligned}$$

$$\begin{aligned}\sigma_{A\ominus C}(v) &= \inf_{u\in\mathbb{U}} \max\{\sigma_A(v+u), 1-\sigma_C(u)\}\\ &\leq \inf_{u\in\mathbb{U}} \max\{\sigma_B(v+u), 1-\sigma_C(u)\}\\ &= \sigma_{B\ominus C}(v)\end{aligned}$$

$$\begin{aligned}\omega_{A\ominus C}(v) &= \sup_{u\in\mathbb{U}} \min\{\omega_A(v+u), \omega_C(u)\}\\ &\geq \sup_{u\in\mathbb{U}} \min\{\omega_B(v+u), \omega_C(u)\}\\ &= \omega_{B\ominus C}(v)\end{aligned}$$

Quindi $\widetilde{A}\widetilde{\ominus}\widetilde{C} \Subset \widetilde{B}\widetilde{\ominus}\widetilde{C}$.
(2) Dall'ipotesi $\widetilde{A} \Subset \widetilde{B}$, per definizione, si ha che $\mu_A(u) \leq \mu_B(u)$, $\sigma_A(u) \leq \sigma_B(u)$, $\omega_A(u) \geq \omega_B(u)$. Per ogni $v \in \mathbb{U}$, si ha:

$$\begin{aligned}\mu_{C\ominus B}(v) &= \inf_{u\in\mathbb{U}} \max\{\mu_C(x+u), 1-\mu_B(u)\}\\ &\leq \inf_{u\in\mathbb{U}} \max\{\mu_C(x+u), 1-\mu_A(u)\}\\ &= \mu_{C\ominus A}(v)\end{aligned}$$

$$\begin{aligned}\sigma_{C\ominus B}(v) &= \inf_{u\in\mathbb{U}} \max\{\sigma_C(x+u), 1-\sigma_B(u)\}\\ &\leq \inf_{u\in\mathbb{U}} \max\{\sigma_C(x+u), 1-\sigma_A(u)\}\\ &= \sigma_{C\ominus A}(v)\end{aligned}$$

$$\begin{aligned}\omega_{C\ominus B}(v) &= \inf_{u\in\mathbb{U}} \max\{\omega_C(x+u), \omega_B(u)\}\\ &\geq \inf_{u\in\mathbb{U}} \max\{\omega_C(x+u), \omega_A(u)\}\\ &= \omega_{C\ominus A}(v)\end{aligned}$$

Quindi $\widetilde{C}\widetilde{\ominus}\widetilde{B} \Subset \widetilde{C}\widetilde{\ominus}\widetilde{A}$. □

Proposizione 4.2.11. [11] [12] *Siano* $\left\{\widetilde{A}_i\right\}_{i\in I}$ *una famiglia di SVN-set* $\widetilde{A}_i = \langle \mathbb{U}, \mu_{A_i}, \sigma_{A_i}, \omega_{A_i}\rangle$ *e* $\widetilde{B} = \langle \mathbb{U}, \mu_B, \sigma_B, \omega_B\rangle$ *un SVN-set su uno stesso insieme universo* $\mathbb{U}$*, allora, si ha che:*

(1) $\left(\bigcap_{i\in I} \widetilde{A}_i\right) \widetilde{\ominus} \widetilde{B} \Subset \bigcap_{i\in I} (\widetilde{A}_i \widetilde{\ominus} \widetilde{B})$

(2) $\bigcup_{i\in I} (\widetilde{A}_i \widetilde{\ominus} \widetilde{B}) \Subset \left(\bigcup_{i\in I} \widetilde{A}_i\right) \widetilde{\ominus} \widetilde{B}$

Dimostrazione. (1) Per ogni $v \in \mathbb{U}$, si ha:

$$\begin{aligned}\mu_{(\bigcap_{i\in I} A_i)\ominus B}(v) &= \inf_{u\in\mathbb{U}} \max\left\{\mu_{(\bigcap_{i\in I} A_i)}(v+u), 1-\mu_B(u)\right\}\\ &= \inf_{u\in\mathbb{U}} \max\left\{\inf_{i\in I} \mu_{A_i}(v+u), 1-\mu_B(u)\right\}\\ &\leq \inf_{u\in\mathbb{U}} \inf_{i\in I} \max\{\mu_{A_i}(v+u), 1-\mu_B(u)\}\\ &= \bigwedge_{i\in I} \inf_{u\in\mathbb{U}} \max\{\mu_{A_i}(v+u), 1-\mu_B(u)\}\\ &= \bigwedge_{i\in I} \mu_{A_i\ominus B}(v)\\ &= \mu_{\bigcap_{i\in I}(A_i\ominus B)}(v)\end{aligned}$$

$$\begin{aligned}
\sigma_{(\mathbb{m}_{i\in I}A_i)\ominus B}(v) &= \inf_{u\in\mathbb{U}} \max\left\{\sigma_{(\mathbb{m}_{i\in I}A_i)}(v+u), 1-\sigma_B(u)\right\}\\
&= \inf_{u\in\mathbb{U}} \max\left\{\inf_{i\in I}\sigma_{A_i}(v+u), 1-\sigma_B(u)\right\}\\
&\leq \inf_{u\in\mathbb{U}} \inf_{i\in I} \max\left\{\sigma_{A_i}(v+u), 1-\sigma_B(u)\right\}\\
&= \bigwedge_{i\in I} \inf_{u\in\mathbb{U}} \max\left\{\sigma_{A_i}(v+u), 1-\sigma_B(u)\right\}\\
&= \bigwedge_{i\in I} \sigma_{A_i\widetilde{\ominus}B}(v)\\
&= \sigma_{\mathbb{m}_{i\in I}(A_i\ominus B)}(v)
\end{aligned}$$

$$\begin{aligned}
\omega_{(\mathbb{m}_{i\in I}A_i)\ominus B}(v) &= \sup_{u\in\mathbb{U}} \min\left\{\omega_{(\mathbb{m}_{i\in I}A_i)}(v+u), \omega_B(u)\right\}\\
&= \sup_{u\in\mathbb{U}} \min\left\{\sup_{i\in I}\omega_{A_i}(v+u), \omega_B(u)\right\}\\
&\geq \sup_{u\in\mathbb{U}} \sup_{i\in I} \min\left\{\omega_{A_i}(v+u), \omega_B(u)\right\}\\
&= \bigvee_{i\in I} \sup_{u\in\mathbb{U}} \min\left\{\omega_{A_i}(v+u), \omega_B(u)\right\}\\
&= \bigvee_{i\in I} \omega_{A_i\ominus B}(v)\\
&= \omega_{\mathbb{m}_{i\in I}(A_i\ominus B)}(v)
\end{aligned}$$

Dunque, essendo

$$\mu_{(\mathbb{m}_{i\in I}A_i)\ominus B}(v) \leq \mu_{\mathbb{m}_{i\in I}(A_i\ominus B)}(v),$$
$$\sigma_{(\mathbb{m}_{i\in I}A_i)\ominus B}(v) \leq \sigma_{\mathbb{m}_{i\in I}(A_i\ominus B)}(v)$$

e

$$\omega_{(\mathbb{m}_{i\in I}A_i)\ominus B}(v) \geq \omega_{\mathbb{m}_{i\in I}(A_i\ominus B)}(v)$$

per ogni $v \in \mathbb{U}$, resta provato che $\left(\bigmathbb{M}_{i\in I}\widetilde{A}_i\right)\widetilde{\ominus}\widetilde{B} \Subset \bigmathbb{M}_{i\in I}(\widetilde{A}_i\widetilde{\ominus}\widetilde{B})$.
(2) Per ogni $v \in \mathbb{U}$, si ha:

$$\begin{aligned}
\mu_{\mathbb{u}_{i\in I}(A_i\ominus B)}(v) &= \sup_{i\in I} \mu_{A_i\ominus B}(v)\\
&= \sup_{i\in I} \inf_{u\in\mathbb{U}} \max\left\{\mu_{A_i}(v+u), \mu_B(u)\right\}\\
&= \inf_{u\in\mathbb{U}} \sup_{i\in I} \max\left\{\mu_{A_i}(v+u), \mu_B(u)\right\}\\
&\leq \inf_{u\in\mathbb{U}} \max\left\{\sup_{i\in I}\mu_{A_i}(v+u), \mu_B(u)\right\}\\
&= \inf_{u\in\mathbb{U}} \max\left\{\mu_{(\mathbb{u}_{i\in I}A_i)}(v+u), \mu_B(u)\right\}\\
&= \mu_{(\mathbb{u}_{i\in I}A_i)\ominus B}(v)
\end{aligned}$$

$$
\begin{aligned}
\sigma_{\Cup_{i\in I}(A_i\ominus B)}(v) &= \sup_{i\in I}\sigma_{A_i\ominus B}(v)\\
&= \sup_{i\in I}\inf_{u\in\mathbb{U}}\max\{\sigma_{A_i}(v+u),\sigma_B(u)\}\\
&= \inf_{u\in\mathbb{U}}\sup_{i\in I}\max\{\sigma_{A_i}(v+u),\sigma_B(u)\}\\
&\leq \inf_{u\in\mathbb{U}}\max\left\{\sup_{i\in I}\sigma_{A_i}(v+u),\sigma_B(u)\right\}\\
&= \inf_{u\in\mathbb{U}}\max\left\{\sigma_{(\Cup_{i\in I}A_i)}(v+u),\sigma_B(u)\right\}\\
&= \sigma_{(\Cup_{i\in I}A_i)\ominus B}(v)
\end{aligned}
$$

$$
\begin{aligned}
\omega_{\Cup_{i\in I}(A_i\ominus B)}(v) &= \inf_{i\in I}\omega_{A_i\ominus B}(v)\\
&= \inf_{i\in I}\sup_{u\in\mathbb{U}}\min\{\omega_{A_i}(v+u),\omega_B(u)\}\\
&= \sup_{u\in\mathbb{U}}\inf_{i\in I}\min\{\omega_{A_i}(v+u),\omega_B(u)\}\\
&\geq \sup_{u\in\mathbb{U}}\min\left\{\inf_{i\in I}\omega_{A_i}(v+u),\omega_B(u)\right\}\\
&= \sup_{u\in\mathbb{U}}\min\left\{\omega_{(\Cup_{i\in I}A_i)}(v+u),\omega_B(u)\right\}\\
&= \omega_{(\Cup_{i\in I}A_i)\ominus B}(v)
\end{aligned}
$$

Dunque, dal momento che

$$\mu_{\Cup_{i\in I}(A_i\ominus B)}(v) \leq \mu_{(\Cup_{i\in I}A_i)\ominus B}(v),$$

$$\sigma_{\Cup_{i\in I}(A_i\ominus B)}(v) \leq \sigma_{(\Cup_{i\in I}A_i)\ominus B}(v)$$

e

$$\omega_{\Cup_{i\in I}(A_i\ominus B)}(v) \geq \omega_{(\Cup_{i\in I}A_i)\ominus B}(v)$$

per ogni $v\in\mathbb{U}$, resta provata la tesi, cioè che $\bigcup_{i\in I}(\widetilde{A}_i\widetilde{\ominus}\widetilde{B}) \Subset \left(\bigcup_{i\in I}\widetilde{A}_i\right)\widetilde{\ominus}\widetilde{B}$. □

4.2.3 Apertura Neutrosofica

L'operatore di apertura neutrosofica costituisce una generalizzazione del caso binario, tant'è che possiamo definirlo come composizione tra erosione neutrosofica e dilatazione neutrosofica, ovvero:

Definizione 4.2.12. [12] *Siano $\widetilde{A}=\langle\mathbb{U},\mu_A,\sigma_A,\omega_A\rangle$ e $\widetilde{B}=\langle\mathbb{U},\mu_B,\sigma_B,\omega_B\rangle$ due SVN-set su uno stesso insieme universo $\mathbb{U}=\mathbb{Z}^2$. Definiamo l'**apertura neutrosofica** dell'insieme neutrosofico $\widetilde{A}$ rispetto all'elemento strutturante $\widetilde{B}$ come la dilatazione neutrosofica dell'erosione neutrosofica di $\widetilde{A}$ rispetto allo stesso elemento strutturante $\widetilde{B}$, cioè:*

$$A\widetilde{\circ}B=(\widetilde{A}\widetilde{\ominus}\widetilde{B})\widetilde{\oplus}\widetilde{B}=\langle\mathbb{U},\mu_{A\circ B},\sigma_{A\circ B},\omega_{A\circ B}\rangle$$

dove le funzioni di appartenenza, indeterminatezza e non appartenenza sono definite rispettivamente per ogni $v \in \mathbb{U}$ da:

$$\mu_{A\circ B}(v) = \sup_{u\in\mathbb{U}} \min\left\{ \inf_{w\in\mathbb{U}} \max\left\{\mu_A(v-u+w), 1-\mu_B(w)\right\}, \mu_B(u)\right\}$$
$$\sigma_{A\circ B}(v) = \sup_{u\in\mathbb{U}} \min\left\{ \inf_{w\in\mathbb{U}} \max\left\{\sigma_A(v-u+w), 1-\sigma_B(w)\right\}, \sigma_B(u)\right\}$$
$$\omega_{A\circ B}(v) = \inf_{u\in\mathbb{U}} \max\left\{ \sup_{w\in\mathbb{U}} \min\left\{\omega_A(v-u+w), \omega_B(w)\right\}, 1-\omega_B(u)\right\}$$

Proposizione 4.2.13. [11] *Siano $\widetilde{A} = \langle \mathbb{U}, \mu_A, \sigma_A, \omega_A\rangle$, $\widetilde{B} = \langle \mathbb{U}, \mu_B, \sigma_B, \omega_B\rangle$, $\widetilde{C} = \langle \mathbb{U}, \mu_C, \sigma_C, \omega_C\rangle$ SVN-set in $\mathcal{SVN}(\mathbb{U})$ tali che $\widetilde{A} \Subset \widetilde{B}$. Allora $\widetilde{A}\widetilde{\circ}\widetilde{C} \Subset \widetilde{B}\widetilde{\circ}\widetilde{C}$.*

Dimostrazione. Dall'ipotesi $\widetilde{A} \Subset \widetilde{B}$, per definizione, si ha che $\mu_A(u) \leq \mu_B(u)$, $\sigma_A(u) \leq \sigma_B(u)$, $\omega_A(u) \geq \omega_B(u)$. Per ogni $v \in \mathbb{U}$, allora si ha:

$$\begin{aligned}\mu_{A\circ C}(v) &= \sup_{u\in\mathbb{U}} \min\left\{ \inf_{w\in\mathbb{U}} \max\left\{\mu_A(v-u+w), 1-\mu_C(w)\right\}, \mu_C(u)\right\}\\ &\leq \sup_{u\in\mathbb{U}} \min\left\{ \inf_{w\in\mathbb{U}} \max\left\{\mu_B(v-u+w), 1-\mu_C(w)\right\}, \mu_C(u)\right\}\\ &= \mu_{B\circ C}(v)\end{aligned}$$

$$\begin{aligned}\sigma_{A\circ C}(v) &= \sup_{u\in\mathbb{U}} \min\left\{ \inf_{w\in\mathbb{U}} \max\left\{\sigma_A(v-u+w), 1-\sigma_C(w)\right\}, \sigma_C(u)\right\}\\ &\leq \sup_{u\in\mathbb{U}} \min\left\{ \inf_{w\in\mathbb{U}} \max\left\{\sigma_B(v-u+w), 1-\sigma_C(w)\right\}, \sigma_C(u)\right\}\\ &= \sigma_{B\circ C}(v)\end{aligned}$$

$$\begin{aligned}\omega_{A\circ C}(v) &= \inf_{u\in\mathbb{U}} \max\left\{ \sup_{w\in\mathbb{U}} \min\left\{\omega_A(v-u+w), \omega_C(w)\right\}, 1-\omega_C(u)\right\}\\ &\geq \inf_{u\in\mathbb{U}} \max\left\{ \sup_{w\in\mathbb{U}} \min\left\{\omega_B(v-u+w), \omega_C(w)\right\}, 1-\omega_C(u)\right\}\\ &= \omega_{B\circ C}(v)\end{aligned}$$

Quindi $\widetilde{A}\widetilde{\circ}\widetilde{C} \Subset \widetilde{B}\widetilde{\circ}\widetilde{C}$. □

Proposizione 4.2.14. [11] [12]*Siano $\left\{\widetilde{A}_i\right\}_{i\in I}$ una famiglia di SVN-set $\widetilde{A}_i = \langle \mathbb{U}, \mu_{A_i}, \sigma_{A_i}, \omega_{A_i}\rangle$ e $\widetilde{B} = \langle \mathbb{U}, \mu_B, \sigma_B, \omega_B\rangle$ un SVN-set su uno stesso insieme universo $\mathbb{U}$, allora, si ha che:*

(1) $\left(\bigcap_{i\in I} \widetilde{A}_i\right)\widetilde{\circ}\widetilde{B} \Subset \bigcap_{i\in I}(\widetilde{A}_i\widetilde{\circ}\widetilde{B})$

(2) $\bigcup_{i\in I}(\widetilde{A}_i\widetilde{\circ}\widetilde{B}) \Subset \left(\bigcup_{i\in I} \widetilde{A}_i\right)\widetilde{\circ}\widetilde{B}$

Dimostrazione. (1) Per ogni $v \in \mathbb{U}$ si ha che:

$$\begin{aligned}
\mu_{(\mathbb{m}_{i\in I}A_i)\circ B}(v) &= \sup_{u\in\mathbb{U}} \min\left\{ \inf_{w\in\mathbb{U}} \max\left\{\mu_{(\mathbb{m}_{i\in I}A_i)}(v-u+w), 1-\mu_B(w)\right\}, \mu_B(u)\right\} \\
&= \sup_{u\in\mathbb{U}} \min\left\{ \inf_{w\in\mathbb{U}} \max\left\{\inf_{i\in I}\mu_{A_i}(v-u+w), 1-\mu_B(w)\right\}, \mu_B(u)\right\} \\
&\leq \sup_{u\in\mathbb{U}} \min\left\{ \inf_{w\in\mathbb{U}} \inf_{i\in I}\max\left\{\mu_{A_i}(v-u+w), 1-\mu_B(w)\right\}, \mu_B(u)\right\} \\
&\leq \inf_{i\in I}\sup_{u\in\mathbb{U}} \min\left\{ \inf_{w\in\mathbb{U}} \max\left\{\mu_{A_i}(v-u+w), 1-\mu_B(w)\right\}, \mu_B(u)\right\} \\
&= \bigwedge_{i\in I} \mu_{A_i\circ B}(v) \\
&= \mu_{\mathbb{m}_{i\in I}(A_i\circ B)}(v)
\end{aligned}$$

$$\begin{aligned}
\sigma_{(\mathbb{m}_{i\in I}A_i)\circ B}(v) &= \sup_{u\in\mathbb{U}} \min\left\{ \inf_{w\in\mathbb{U}} \max\left\{\sigma_{(\mathbb{m}_{i\in I}A_i)}(v-u+w), 1-\sigma_B(w)\right\}, \sigma_B(u)\right\} \\
&= \sup_{u\in\mathbb{U}} \min\left\{ \inf_{w\in\mathbb{U}} \max\left\{\inf_{i\in I}\sigma_{A_i}(v-u+w), 1-\sigma_B(w)\right\}, \sigma_B(u)\right\} \\
&\leq \sup_{u\in\mathbb{U}} \min\left\{ \inf_{w\in\mathbb{U}} \inf_{i\in I}\max\left\{\sigma_{A_i}(v-u+w), 1-\sigma_B(w)\right\}, \sigma_B(u)\right\} \\
&\leq \inf_{i\in I}\sup_{u\in\mathbb{U}} \min\left\{ \inf_{w\in\mathbb{U}} \max\left\{\sigma_{A_i}(v-u+w), 1-\sigma_B(w)\right\}, \sigma_B(u)\right\} \\
&= \bigwedge_{i\in I} \sigma_{A_i\circ B}(v) \\
&= \sigma_{\mathbb{m}_{i\in I}(A_i\circ B)}(v)
\end{aligned}$$

$$\begin{aligned}
\omega_{(\mathbb{m}_{i\in I}A_i)\circ B}(v) &= \inf_{u\in\mathbb{U}} \max\left\{ \sup_{w\in\mathbb{U}} \min\left\{\omega_{(\mathbb{m}_{i\in I}A_i)}(v-u+w), \omega_B(w)\right\}, 1-\omega_B(u)\right\} \\
&= \inf_{u\in\mathbb{U}} \max\left\{ \sup_{w\in\mathbb{U}} \min\left\{\sup_{i\in I}\omega_{A_i}(v-u+w), \omega_B(w)\right\}, 1-\omega_B(u)\right\} \\
&\geq \inf_{u\in\mathbb{U}} \max\left\{ \sup_{w\in\mathbb{U}} \sup_{i\in I}\min\left\{\omega_{A_i}(v-u+w), \omega_B(w)\right\}, 1-\omega_B(u)\right\} \\
&\geq \sup_{i\in I}\inf_{u\in\mathbb{U}} \max\left\{ \sup_{w\in\mathbb{U}} \min\left\{\omega_{A_i}(v-u+w), \omega_B(w)\right\}, 1-\omega_B(u)\right\} \\
&= \bigvee_{i\in I} \omega_{A_i\circ B}(v) \\
&= \omega_{\mathbb{m}_{i\in I}(A_i\circ B)}(v)
\end{aligned}$$

Dunque, essendo

$$\mu_{(\mathbb{m}_{i\in I}A_i)\circ B}(v) \leq \mu_{\mathbb{m}_{i\in I}(A_i\circ B)}(v),$$
$$\sigma_{(\mathbb{m}_{i\in I}A_i)\circ B}(v) \leq \sigma_{\mathbb{m}_{i\in I}(A_i\circ B)}(v)$$

e

$$\omega_{(\mathbb{m}_{i\in I}A_i)\circ B}(v) \geq \omega_{\mathbb{m}_{i\in I}(A_i\circ B)}(v)$$

resta provato che $\left(\bigcap_{i\in I}\widetilde{A}_i\right)\widetilde{\circ}\widetilde{B} \Subset \bigcap_{i\in I}(\widetilde{A}_i\widetilde{\circ}\widetilde{B})$.
(2) Per ogni $v \in \mathbb{U}$ si ha che:

$$\begin{aligned}
\mu_{\mathbb{U}_{i\in I}(A_i\circ B)}(v) &= \sup_{i\in I}\mu_{A_i\circ B}(v)\\
&= \sup_{i\in I}\sup_{u\in\mathbb{U}}\min\left\{\inf_{w\in\mathbb{U}}\max\left\{\mu_{A_i}(v-u+w),1-\mu_B(w)\right\},\mu_B(u)\right\}\\
&\leq \sup_{u\in\mathbb{U}}\min\left\{\inf_{w\in\mathbb{U}}\sup_{i\in I}\max\left\{\mu_{A_i}(v-u+w),1-\mu_B(w)\right\},\mu_B(u)\right\}\\
&\leq \sup_{u\in\mathbb{U}}\min\left\{\inf_{w\in\mathbb{U}}\max\left\{\sup_{i\in I}\mu_{A_i}(v-u+w),1-\mu_B(w)\right\},\mu_B(u)\right\}\\
&= \sup_{u\in\mathbb{U}}\min\left\{\inf_{w\in\mathbb{U}}\max\left\{\mu_{(\mathbb{U}_{i\in I}A_i)}(v-u+w),1-\mu_B(w)\right\},\mu_B(u)\right\}\\
&= \mu_{(\mathbb{U}_{i\in I}A_i)\circ B}(v)
\end{aligned}$$

$$\begin{aligned}
\sigma_{\mathbb{U}_{i\in I}(A_i\circ B)}(v) &= \sup_{i\in I}\sigma_{A_i\circ B}(v)\\
&= \sup_{i\in I}\sup_{u\in\mathbb{U}}\min\left\{\inf_{w\in\mathbb{U}}\max\left\{\sigma_{A_i}(v-u+w),1-\sigma_B(w)\right\},\sigma_B(u)\right\}\\
&\leq \sup_{u\in\mathbb{U}}\min\left\{\inf_{w\in\mathbb{U}}\sup_{i\in I}\max\left\{\sigma_{A_i}(v-u+w),1-\sigma_B(w)\right\},\sigma_B(u)\right\}\\
&\leq \sup_{u\in\mathbb{U}}\min\left\{\inf_{w\in\mathbb{U}}\max\left\{\sup_{i\in I}\sigma_{A_i}(v-u+w),1-\sigma_B(w)\right\},\sigma_B(u)\right\}\\
&= \sup_{u\in\mathbb{U}}\min\left\{\inf_{w\in\mathbb{U}}\max\left\{\sigma_{(\mathbb{U}_{i\in I}A_i)}(v-u+w),1-\sigma_B(w)\right\},\sigma_B(u)\right\}\\
&= \sigma_{(\mathbb{U}_{i\in I}A_i)\circ B}(v)
\end{aligned}$$

$$\begin{aligned}
\omega_{\mathbb{U}_{i\in I}(A_i\circ B)}(v) &= \inf_{i\in I}\omega_{A_i\circ B}(v)\\
&= \inf_{i\in I}\inf_{u\in\mathbb{U}}\max\left\{\sup_{w\in\mathbb{U}}\min\left\{\omega_{A_i}(v-u+w),\omega_B(w)\right\},1-\omega_B(u)\right\}\\
&\geq \inf_{u\in\mathbb{U}}\max\left\{\sup_{w\in\mathbb{U}}\inf_{i\in I}\min\left\{\omega_{A_i}(v-u+w),\omega_B(w)\right\},1-\omega_B(u)\right\}\\
&\geq \inf_{u\in\mathbb{U}}\max\left\{\sup_{w\in\mathbb{U}}\min\left\{\inf_{i\in I}\omega_{A_i}(v-u+w),\omega_B(w)\right\},1-\omega_B(u)\right\}\\
&= \inf_{u\in\mathbb{U}}\max\left\{\sup_{w\in\mathbb{U}}\min\left\{\omega_{(\mathbb{U}_{i\in I}A_i)}(v-u+w),\omega_B(w)\right\},1-\omega_B(u)\right\}\\
&= \omega_{(\mathbb{U}_{i\in I}A_i)\circ B}(v)
\end{aligned}$$

Quindi, dal momento che

$$\mu_{\mathbb{U}_{i\in I}(A_i\circ B)}(v) \leq \mu_{(\mathbb{U}_{i\in I}A_i)\circ B}(v),$$
$$\sigma_{\mathbb{U}_{i\in I}(A_i\circ B)}(v) \leq \sigma_{(\mathbb{U}_{i\in I}A_i)\circ B}(v)$$

e

$$\omega_{\mathbb{U}_{i\in I}(A_i\circ B)}(v) \geq \omega_{(\mathbb{U}_{i\in I}A_i)\circ B}(v)$$

per ogni $v \in \mathbb{U}$, resta provata l'inclusione $\bigcup_{i\in I}(\widetilde{A}_i\widetilde{\circ}\widetilde{B}) \Subset \left(\bigcup_{i\in I}\widetilde{A}_i\right)\widetilde{\circ}\widetilde{B}$. □

4.2.4 Chiusura Neutrosofica

L'operatore di chiusura neutrosofica costituisce una generalizzazione del caso binario, tant'è che possiamo definirlo come composizione tra dilatazione neutrosofica ed erosione neutrosofica, ovvero:

Definizione 4.2.15. [12] *Siano $\widetilde{A} = \langle \mathbb{U}, \mu_A, \sigma_A, \omega_A \rangle$ e $\widetilde{B} = \langle \mathbb{U}, \mu_B, \sigma_B, \omega_B \rangle$ due SVN-set su uno stesso insieme universo $\mathbb{U} = \mathbb{Z}^2$. Definiamo la* ***chiusura neutrosofica*** *dell'insieme neutrosofico $\widetilde{A}$ rispetto all'elemento strutturante $\widetilde{B}$ come l'erosione neutrosofica della dilatazione neutrosofica di $\widetilde{A}$ rispetto allo stesso elemento strutturante $\widetilde{B}$, cioè:*

$$\widetilde{A} \widetilde{\bullet} \widetilde{B} = (\widetilde{A} \widetilde{\oplus} \widetilde{B}) \widetilde{\ominus} \widetilde{B} = \langle \mathbb{U}, \mu_{A \bullet B}, \sigma_{A \bullet B}, \omega_{A \bullet B} \rangle$$

dove le funzioni di appartenenza, indeterminatezza e non appartenenza sono definite rispettivamente per ogni $v \in \mathbb{U}$ da:

$$\mu_{A \bullet B}(v) = \inf_{u \in \mathbb{U}} \max \left\{ \sup_{w \in \mathbb{U}} \min \left\{ \mu_A(v - u + w), \mu_B(w) \right\}, 1 - \mu_B(u) \right\}$$

$$\sigma_{A \bullet B}(v) = \inf_{u \in \mathbb{U}} \max \left\{ \sup_{w \in \mathbb{U}} \min \left\{ \sigma_A(v - u + w), \sigma_B(w) \right\}, 1 - \sigma_B(u) \right\}$$

$$\omega_{A \bullet B}(v) = \sup_{u \in \mathbb{U}} \min \left\{ \inf_{w \in \mathbb{U}} \max \left\{ \omega_A(v - u + w), 1 - \omega_B(w) \right\}, \omega_B(u) \right\}$$

Proposizione 4.2.16. [12] *Siano $\widetilde{A} = \langle \mathbb{U}, \mu_A, \sigma_A, \omega_A \rangle$ e $\widetilde{B} = \langle \mathbb{U}, \mu_B, \sigma_B, \omega_B \rangle$ due SVN-set su uno stesso insieme universo $\mathbb{U}$, allora l'apertura e la chiusura neutrosofica sono duali, cioè:*

$$(\widetilde{A}^{\complement} \widetilde{\bullet} \widetilde{B})^{\complement} = \widetilde{A} \widetilde{\circ} \widetilde{B}$$

Dimostrazione. Per definizione, $(\widetilde{A}^{\complement} \widetilde{\bullet} \widetilde{B})^{\complement} = \left\langle \mathbb{U}, \mu_{(A^{\complement} \bullet B)^{\complement}}, \sigma_{(A^{\complement} \bullet B)^{\complement}}, \omega_{(A^{\complement} \bullet B)^{\complement}} \right\rangle$, dove per ogni $v \in \mathbb{U}$ si ha che:

$$\begin{aligned}
\mu_{(A^{\complement} \bullet B)^{\complement}}(v) &= 1 - \mu_{A^{\complement} \bullet B}(v) \\
&= 1 - \inf_{u \in \mathbb{U}} \max \left\{ \sup_{w \in \mathbb{U}} \min \left\{ \mu_{A^{\complement}}(v - u + w), \mu_B(w) \right\}, 1 - \mu_B(u) \right\} \\
&= \sup_{u \in \mathbb{U}} \min \left\{ 1 - \sup_{w \in \mathbb{U}} \min \left\{ \mu_{A^{\complement}}(v - u + w), \mu_B(w) \right\}, \mu_B(u) \right\} \\
&= \sup_{u \in \mathbb{U}} \min \left\{ \inf_{w \in \mathbb{U}} \max \left\{ 1 - \mu_{A^{\complement}}(v - u + w), 1 - \mu_B(w) \right\}, \mu_B(u) \right\} \\
&= \sup_{u \in \mathbb{U}} \min \left\{ \inf_{w \in \mathbb{U}} \max \left\{ \mu_A(v - u + w), 1 - \mu_B(w) \right\}, \mu_B(u) \right\} \\
&= \mu_{A \circ B}(v)
\end{aligned}$$

$$
\begin{aligned}
\sigma_{(A^{\complement}\bullet B)^{\complement}}(v) &= 1-\sigma_{A^{\complement}\bullet B}(v)\\
&= 1-\inf_{u\in\mathbb{U}}\max\left\{\sup_{w\in\mathbb{U}}\min\left\{\sigma_{A^{\complement}}(v-u+w),\sigma_B(w)\right\},1-\sigma_B(u)\right\}\\
&= \sup_{u\in\mathbb{U}}\min\left\{1-\sup_{w\in\mathbb{U}}\min\left\{\sigma_{A^{\complement}}(v-u+w),\sigma_B(w)\right\},\sigma_B(u)\right\}\\
&= \sup_{u\in\mathbb{U}}\min\left\{\inf_{w\in\mathbb{U}}\max\left\{1-\sigma_{A^{\complement}}(v-u+w),1-\sigma_B(w)\right\},\sigma_B(u)\right\}\\
&= \sup_{u\in\mathbb{U}}\min\left\{\inf_{w\in\mathbb{U}}\max\left\{\sigma_A(v-u+w),1-\sigma_B(w)\right\},\sigma_B(u)\right\}\\
&= \sigma_{A\circ B}(v)
\end{aligned}
$$

$$
\begin{aligned}
\omega_{(A^{\complement}\bullet B)^{\complement}}(v) &= 1-\omega_{A^{\complement}\bullet B}(v)\\
&= 1-\sup_{u\in\mathbb{U}}\min\left\{\inf_{w\in\mathbb{U}}\max\left\{\omega_{A^{\complement}}(v-u+w),1-\omega_B(w)\right\},\omega_B(u)\right\}\\
&= \inf_{u\in\mathbb{U}}\max\left\{1-\inf_{w\in\mathbb{U}}\max\left\{\omega_{A^{\complement}}(v-u+w),1-\omega_B(w)\right\},1-\omega_B(u)\right\}\\
&= \inf_{u\in\mathbb{U}}\max\left\{\sup_{w\in\mathbb{U}}\min\left\{\omega_A(v-u+w),\omega_B(w)\right\},1-\omega_B(u)\right\}\\
&= \omega_{A\circ B}(v)
\end{aligned}
$$

Dunque $\left\langle \mathbb{U},\mu_{(A^C\bullet B)^C},\sigma_{(A^C\bullet B)^C},\omega_{(A^C\bullet B)^C}\right\rangle = \langle \mathbb{U},\mu_{A\circ B},\sigma_{A\circ B},\omega_{A\circ B}\rangle$.
Quindi la tesi è provata. □

Proprietà 6. [11] *Siano* $\widetilde{A}=\langle \mathbb{U},\mu_A,\sigma_A,\omega_A\rangle$, $\widetilde{B}=\langle \mathbb{U},\mu_B,\sigma_B,\omega_B\rangle$, $\widetilde{C}=\langle \mathbb{U},\mu_C,\sigma_C,\omega_C\rangle$ *SVN-set in* $\mathcal{SVN}(\mathbb{U})$ *tali che* $\widetilde{A}\Subset\widetilde{B}$. *Allora* $\widetilde{A}\widetilde{\bullet}\widetilde{C}\Subset\widetilde{B}\widetilde{\bullet}\widetilde{C}$.

Dimostrazione. Dall'ipotesi $\widetilde{A}\Subset\widetilde{B}$, per definizione, si ha che $\mu_A(u)\leq\mu_B(u)$, $\sigma_A(u)\leq\sigma_B(u)$, $\omega_A(u)\geq\omega_B(u)$. Per ogni $v\in\mathbb{U}$, allora si ha:

$$
\begin{aligned}
\mu_{A\bullet C}(v) &= \inf_{u\in\mathbb{U}}\max\left\{\sup_{w\in\mathbb{U}}\min\left\{\mu_A(v-u+w),\mu_C(w)\right\},1-\mu_C(u)\right\}\\
&\leq \inf_{u\in\mathbb{U}}\max\left\{\sup_{w\in\mathbb{U}}\min\left\{\mu_B(v-u+w),\mu_C(w)\right\},1-\mu_C(u)\right\}\\
&= \mu_{B\bullet C}(v)
\end{aligned}
$$

$$
\begin{aligned}
\sigma_{A\bullet C}(v) &= \inf_{u\in\mathbb{U}}\max\left\{\sup_{w\in\mathbb{U}}\min\left\{\sigma_A(v-u+w),\sigma_C(w)\right\},1-\sigma_C(u)\right\}\\
&\leq \inf_{u\in\mathbb{U}}\max\left\{\sup_{w\in\mathbb{U}}\min\left\{\sigma_B(v-u+w),\sigma_C(w)\right\},1-\sigma_C(u)\right\}\\
&= \sigma_{B\bullet C}(v)
\end{aligned}
$$

$$
\begin{aligned}
\omega_{A\bullet C}(v) &= \sup_{u\in\mathbb{U}}\min\left\{\inf_{w\in\mathbb{U}}\max\left\{\omega_A(v-u+w),1-\omega_C(w)\right\},\omega_C(u)\right\}\\
&\geq \sup_{u\in\mathbb{U}}\min\left\{\inf_{w\in\mathbb{U}}\max\left\{\omega_B(v-u+w),1-\omega_C(w)\right\},\omega_C(u)\right\}\\
&= \omega_{B\bullet C}(v)
\end{aligned}
$$

□

Proposizione 4.2.17. [11] [12] *Siano* $\left\{\widetilde{A}_i\right\}_{i\in I}$ *una famiglia di SVN-set* $\widetilde{A}_i = \langle \mathbb{U}, \mu_{A_i}, \sigma_{A_i}, \omega_{A_i}\rangle$ *e* $\widetilde{B} = \langle \mathbb{U}, \mu_B, \sigma_B, \omega_B\rangle$ *un SVN-set su uno stesso insieme universo* $\mathbb{U}$*, allora, si ha che:*

(1) $\left(\bigcap_{i\in I} \widetilde{A}_i\right) \widetilde{\bullet} \widetilde{B} \Subset \bigcap_{i\in I} (\widetilde{A}_i \widetilde{\bullet} \widetilde{B})$

(2) $\bigcup_{i\in I} (\widetilde{A}_i \widetilde{\bullet} \widetilde{B}) \Subset \left(\bigcup_{i\in I} \widetilde{A}_i\right) \widetilde{\bullet} \widetilde{B}$

Dimostrazione. (1) Per ogni $v \in \mathbb{U}$ si ha che:

$$\begin{aligned}
\mu_{(\mathbb{\cap}_{i\in I} A_i)\bullet B}(v) &= \inf_{u\in\mathbb{U}} \max\left\{\sup_{w\in\mathbb{U}} \min\left\{\mu_{(\mathbb{\cap} A_i)}(v-u+w), \mu_B(w)\right\}, 1-\mu_B(u)\right\} \\
&= \inf_{u\in\mathbb{U}} \max\left\{\sup_{w\in\mathbb{U}} \min\left\{\inf_{i\in I} \mu_{A_i}(v-u+w), \mu_B(w)\right\}, 1-\mu_B(u)\right\} \\
&\leq \inf_{u\in\mathbb{U}} \max\left\{\sup_{w\in\mathbb{U}} \inf_{i\in I} \min\left\{\mu_{A_i}(v-u+w), \mu_B(w)\right\}, 1-\mu_B(u)\right\} \\
&\leq \inf_{i\in I} \inf_{u\in\mathbb{U}} \max\left\{\sup_{w\in\mathbb{U}} \min\left\{\mu_{A_i}(v-u+w), \mu_B(w)\right\}, 1-\mu_B(u)\right\} \\
&= \bigwedge_{i\in I} \inf_{u\in\mathbb{U}} \max\left\{\sup_{w\in\mathbb{U}} \min\left\{\mu_{A_i}(v-u+w), \mu_B(w)\right\}, 1-\mu_B(u)\right\} \\
&= \bigwedge_{i\in I} \mu_{A_i\bullet B}(v) \\
&= \mu_{\mathbb{\cap}_{i\in I}(A_i\bullet B)}(v)
\end{aligned}$$

$$\begin{aligned}
\sigma_{(\mathbb{\cap}_{i\in I} A_i)\bullet B}(v) &= \inf_{u\in\mathbb{U}} \max\left\{\sup_{w\in\mathbb{U}} \min\left\{\sigma_{(\mathbb{\cap} A_i)}(v-u+w), \sigma_B(w)\right\}, 1-\sigma_B(u)\right\} \\
&= \inf_{u\in\mathbb{U}} \max\left\{\sup_{w\in\mathbb{U}} \min\left\{\inf_{i\in I} \sigma_{A_i}(v-u+w), \sigma_B(w)\right\}, 1-\sigma_B(u)\right\} \\
&\leq \inf_{u\in\mathbb{U}} \max\left\{\sup_{w\in\mathbb{U}} \inf_{i\in I} \min\left\{\sigma_{A_i}(v-u+w), \sigma_B(w)\right\}, 1-\sigma_B(u)\right\} \\
&\leq \inf_{i\in I} \inf_{u\in\mathbb{U}} \max\left\{\sup_{w\in\mathbb{U}} \min\left\{\sigma_{A_i}(v-u+w), \sigma_B(w)\right\}, 1-\sigma_B(u)\right\} \\
&= \bigwedge_{i\in I} \inf_{u\in\mathbb{U}} \max\left\{\sup_{w\in\mathbb{U}} \min\left\{\sigma_{A_i}(v-u+w), \sigma_B(w)\right\}, 1-\sigma_B(u)\right\} \\
&= \bigwedge_{i\in I} \sigma_{A_i\bullet B}(v) \\
&= \sigma_{\mathbb{\cap}_{i\in I}(A_i\bullet B)}(v)
\end{aligned}$$

$$
\begin{aligned}
\omega_{(\mathbb{\Cap}_{i\in I}A_i)\bullet B}(v) &= \sup_{u\in\mathbb{U}} \min\left\{ \inf_{w\in\mathbb{U}} \max\left\{\omega_{(\mathbb{\Cap}_{i\in I}A_i)}(v-u+w), 1-\omega_B(w)\right\}, \omega_B(u)\right\} \\
&= \sup_{u\in\mathbb{U}} \min\left\{ \inf_{w\in\mathbb{U}} \max\left\{\sup_{i\in I}\omega_{A_i}(v-u+w), 1-\omega_B(w)\right\}, \omega_B(u)\right\} \\
&\geq \sup_{u\in\mathbb{U}} \min\left\{ \inf_{w\in\mathbb{U}} \sup_{i\in I}\max\left\{\omega_{A_i}(v-u+w), 1-\omega_B(w)\right\}, \omega_B(u)\right\} \\
&\geq \sup_{i\in I}\sup_{u\in\mathbb{U}} \min\left\{ \inf_{w\in\mathbb{U}} \max\left\{\omega_{A_i}(v-u+w), 1-\omega_B(w)\right\}, \omega_B(u)\right\} \\
&= \bigvee_{i\in I}\sup_{u\in\mathbb{U}} \min\left\{ \inf_{w\in\mathbb{U}} \max\left\{\omega_{A_i}(v-u+w), 1-\omega_B(w)\right\}, \omega_B(u)\right\} \\
&= \bigvee_{i\in I} \omega_{A_i\bullet B}(v) \\
&= \omega_{\mathbb{\Cap}(A_i\bullet B)}(v)
\end{aligned}
$$

Dunque, essendo

$$
\begin{aligned}
\mu_{(\mathbb{\Cap}_{i\in I}A_i)\bullet B}(v) &\leq \mu_{\mathbb{\Cap}_{i\in I}(A_i\bullet B)}(v), \\
\sigma_{(\mathbb{\Cap}_{i\in I}A_i)\bullet B}(v) &\leq \sigma_{\mathbb{\Cap}_{i\in I}(A_i\bullet B)}(v)
\end{aligned}
$$

e

$$
\omega_{(\mathbb{\Cap}_{i\in I}A_i)\bullet B}(v) \geq \omega_{\mathbb{\Cap}(A_i\bullet B)}(v)
$$

per ogni $v\in\mathbb{U}$, resta provato che $\left(\bigCap_{i\in I}\widetilde{A}_i\right)\widetilde{\bullet}\widetilde{B} \Subset \bigCap_{i\in I}(\widetilde{A}_i\widetilde{\bullet}\widetilde{B})$.
(2) Per ogni $v\in\mathbb{U}$ si ha che:

$$
\begin{aligned}
\mu_{\mathbb{\Cup}_{i\in I}(A_i\bullet B)}(v) &= \sup_{i\in I}\mu_{(A_i\bullet B)}(v) \\
&= \sup_{i\in I}\inf_{u\in\mathbb{U}} \max\left\{\sup_{w\in\mathbb{U}}\min\left\{\mu_{A_i}(v-u+w), \mu_B(w)\right\}, 1-\mu_B(u)\right\} \\
&\leq \inf_{u\in\mathbb{U}} \max\left\{\sup_{w\in\mathbb{U}}\sup_{i\in I}\min\left\{\mu_{A_i}(v-u+w), \mu_B(w)\right\}, 1-\mu_B(u)\right\} \\
&\leq \inf_{u\in\mathbb{U}} \max\left\{\sup_{w\in\mathbb{U}}\min\left\{\sup_{i\in I}\mu_{A_i}(v-u+w), \mu_B(w)\right\}, 1-\mu_B(u)\right\} \\
&= \inf_{u\in\mathbb{U}} \max\left\{\sup_{w\in\mathbb{U}}\min\left\{\mu_{(\mathbb{\Cup}_{i\in I}A_i)}(v-u+w), \mu_B(w)\right\}, 1-\mu_B(u)\right\} \\
&= \mu_{(\mathbb{\Cup}_{i\in I}A_i)\bullet B}(v)
\end{aligned}
$$

$$
\begin{aligned}
\sigma_{\mathbb{\Cup}_{i\in I}(A_i\bullet B)}(v) &= \sup_{i\in I}\sigma_{(A_i\bullet B)}(v) \\
&= \sup_{i\in I}\inf_{u\in\mathbb{U}} \max\left\{\sup_{w\in\mathbb{U}}\min\left\{\sigma_{A_i}(v-u+w), \sigma_B(w)\right\}, 1-\sigma_B(u)\right\} \\
&\leq \inf_{u\in\mathbb{U}} \max\left\{\sup_{w\in\mathbb{U}}\sup_{i\in I}\min\left\{\sigma_{A_i}(v-u+w), \sigma_B(w)\right\}, 1-\sigma_B(u)\right\} \\
&\leq \inf_{u\in\mathbb{U}} \max\left\{\sup_{w\in\mathbb{U}}\min\left\{\sup_{i\in I}\sigma_{A_i}(v-u+w), \sigma_B(w)\right\}, 1-\sigma_B(u)\right\} \\
&= \inf_{u\in\mathbb{U}} \max\left\{\sup_{w\in\mathbb{U}}\min\left\{\sigma_{(\mathbb{\Cup}_{i\in I}A_i)}(v-u+w), \sigma_B(w)\right\}, 1-\sigma_B(u)\right\} \\
&= \sigma_{(\mathbb{\Cup}_{i\in I}A_i)\bullet B}(v)
\end{aligned}
$$

$$
\begin{aligned}
\omega_{\Cup_{i\in I}(A_i\bullet B)}(v) &= \inf_{i\in I}\omega_{(A_i\bullet B)}(v)\\
&= \inf_{i\in I}\sup_{u\in\Cup}\min\left\{\inf_{w\in\Cup}\max\left\{\omega_{A_i}(v-u+w),1-\omega_B(w)\right\},\omega_B(u)\right\}\\
&\geq \sup_{u\in\Cup}\min\left\{\inf_{w\in\Cup}\inf_{i\in I}\max\left\{\omega_{A_i}(v-u+w),1-\omega_B(w)\right\},\omega_B(u)\right\}\\
&\geq \sup_{u\in\Cup}\min\left\{\inf_{w\in\Cup}\max\left\{\inf_{i\in I}\omega_{A_i}(v-u+w),1-\omega_B(w)\right\},\omega_B(u)\right\}\\
&= \sup_{u\in\Cup}\min\left\{\inf_{w\in\Cup}\max\left\{\omega_{(\Cup_{i\in I}A_i)}(v-u+w),1-\omega_B(w)\right\},\omega_B(u)\right\}\\
&= \omega_{(\Cup_{i\in I}A_i)\bullet B}(v)
\end{aligned}
$$

Quindi, dal momento che

$$\mu_{\Cup_{i\in I}(A_i\bullet B)}(v)\leq\mu_{(\Cup_{i\in I}A_i)\bullet B}(v),$$

$$\sigma_{\Cup_{i\in I}(A_i\bullet B)}(v)\leq\sigma_{(\Cup_{i\in I}A_i)\bullet B}(v)$$

e

$$\omega_{\Cup_{i\in I}(A_i\bullet B)}(v)\geq\omega_{(\Cup_{i\in I}A_i)\bullet B}(v)$$

per ogni $v\in\Cup$, resta provata l'inclusione $\bigcup_{i\in I}(\widetilde{A}_i\widetilde{\bullet}\widetilde{B})\Subset\left(\bigcup_{i\in I}\widetilde{A}_i\right)\widetilde{\bullet}\widetilde{B}$. □

Capitolo 5

Morfologia Neutrosofica Computazionale

In questo capitolo, sulla base delle nozioni e delle proprietà esposte in precedenza, poniamo le basi per un approccio di tipo computazionale alla Morfologia Neutrosofica. In particolare, descriveremo le strutture dati ed i metodi impiegati per sviluppare una classe scritta nel linguaggio Python che consenta di applicare in maniera automatica le principali operazioni morfologiche neutrosofiche descritte nel capitolo precedente.

5.1 L'ambiente operativo

Alla base dello sviluppo di una classe vi è la scelta del linguaggio di programmazione dal momento che esso è un fattore determinante sulle prestazioni e sulla flessibilità della struttura stessa. La scelta del linguaggio Python e di alcuni dei suoi pacchetti (Numpy, MatplotLib ed openCV-Python) è motivata da vari fattori:

- Sintassi chiara ed espressiva: Python è un linguaggio dalla sintassi semplice ed estremamente leggibile in grado di rendere il codice più semplice ed espressivo. Ovviamente questa sua caratteristica risulta particolarmente utile nel caso di strutture dati complesse quali immagini neutrosofiche che andremo a creare e per le operazioni di Morfologia Neutrosofica che eseguiremo.

- Librerie standard estese: Python offre una vasta gamma di librerie a carattere scientifico e matematico. Questo consente agli sviluppatori di utilizzare molte funzioni già esistenti per implementare algoritmi complessi ottimizzando la prestazione del codice stesso.

- Facile integrazione con altri strumenti: Python è abbastanza versatile. Ciò vuol dire che ha una elevata possibilità di integrarsi facilmente con altri linguaggi di programmazione e con librerie esterne. In contesti scientifici come questo, dove può essere necessario utilizzare delle librerie specializzate o altri strumenti computazionali esistenti, questa caratteristica di Python può risultare particolarmente utile.

- Apprendimento veloce: Python viene considerato come il più accessibile fra i linguaggi di programmazione e questo lo rende il linguaggio più utilizzato ed adottato da studenti, ricercatori e programmatori esordienti soprattutto in ambito scientifico.

Dunque, la scelta di Python come linguaggio di programmazione della classe, non è casuale, ma è dettata dalla consapevolezza che il suo utilizzo rende l'intera struttura più accessibile,

flessibile e potente riducendo al minimo la difficoltà di stesura della classe stessa anche in termini di tempo di sviluppo del codice stesso.

5.2 I moduli utilizzati

La ricchezza di librerie di cui dispone il linguaggio ci ha consentito di ottimizzare il lavoro di sviluppo appoggiandoci a numerose strutture dati e funzioni già disponibili che sono state utilizzate in particolar modo per la lettura di immagini da file, la loro memorizzazione in formato compatto e la visualizzazione grafica a schermo. In particolare, sono stati utilizzati i pacchetti Numpy, MatplotLib e openCV-Python.

5.2.1 Il modulo Numpy

Il modulo NumPy (dal termine inglese *Numerical Python*) è la principale libreria di calcolo scientifico usata per lavorare con gli *array*. Gli *array* sono una struttura dati di fondamentale importanza nei linguaggi di programmazione. Il motivo per il quale si utilizza il modulo NumPy, sta proprio nella struttura dati di tipo array, poiché l'oggetto `ndarray` è molto più veloce (a livello computazionale) delle liste del classico Python, in quanto sfrutta le ottimizzazioni del calcolo vettoriale del processore della macchina. Inoltre, il modulo NumPy fornisce una vasta gamma di strumenti matematici in grado di eseguire calcoli numerici in maniera efficace ed efficiente, definisce e rende disponibili un grande numero di funzioni per operazioni aritmetiche e di algebra lineare ed ottimizza le funzioni di ricerca ed ordinamento dei dati. I principali metodi del modulo Numpy (indicato con l'alias "np") sono elencati di seguito:

- `np.array()`: crea un array a partire da una lista, una tupla o un altro oggetto di tipo array.
- `shape()`: restituisce le dimensioni di un array.
- `reshape()`: permette di modificare la dimensione di un array.
- `np.zeros(shape, dtype = data-type)`: crea una matrice di zeri. Il parametro `shape` richiede un intero o una tupla di interi, mentre il parametro `dtype` richiede il tipo di dato per la matrice, per esempio:
  - `np.float32`: è il tipo di numero decimale a virgola mobile (o `float`) a precisione singola, cioe a 32 bit
  - `np.float64`: è il tipo di numero decimale a virgola mobile (o `float`) a precisione doppia, cioe a 64 bit
  - `np.uint8`: è il tipo numero intero ad 8 bit
  - `np.uint32`: è il tipo numero intero a 32 bit
  - `np.uint64`: è il tipo numero intero a 64 bit
- `np.min(array, axis, out, keepdims, initial, where)` e `np.max (array, axis, out, keepdims, initial, where)`: ritornano rispettivamente il più piccolo elemento ed il più grande elemento di un vettore. Tra tutti gli argomenti del metodo, quello obbligatorio è il primo, gli altri invece sono opzionali. Nello specifico:
  - `array`: è l'array di ingresso

- `axis`: rappresenta l'asse lungo il quale viene calcolato il minimo e il massimo
- `out`: è l'array su cui memorizzare il valore del minimo ed anche del massimo
- `keepdims`: rappresenta l'argomento di tipo booleano per indicare se preservare o meno la dimensione dell'array di partenza
- `initial`: è il minimo valore di un elemento di uscita
- `where`: richiede un array di elementi di tipo booleano per indicare gli elementi da includere nel calcolo del minimo e del massimo I metodi statici `numpy.min()` e `numpy.max()` sono ugualmente accessibili anche applicati come metodi di calsse `min()` e `max()` direttamente ad un oggetto `nparray`, in particolare ad una matrice.

5.2.2 Il modulo MatplotLib

Il modulo MatplotLib è una libreria per la creazione di grafici per Python e la libreria NumPy. Essa fornisce interfacce di programmazione per applicazioni (o API) orientate agli oggetti che permettono di produrre grafici all'interno di applicazioni. In particolare, utilizzeremo l'interfaccia **pyplot** (di solito importata con l'alias "plt") che consente di tracciare grafici in modo semplice e simile al ben noto programma Matlab. Ogni funzione del modulo **matplotlib.pyplot** consente di creare una figura ed anche di apportare delle modifiche ad una figura come, per esempio, tracciare alcune linee in una determinata area della stessa o permettere l'inserimento di etichette, ecc. Tra i principali metodi di Pyplot troviamo:

- `plt.plot(X-list, Y-list)`: è un metodo che serve a tracciare un punto o una linea (curva o spezzata) passandogli due distinte liste (o array) delle ascisse e delle ordinate. Oltre ai due parametri obbligatori già citati, questo metodo ne possiede altri facoltativi, tra cui:
  - `linestyle`: specifica lo stile della linea
  - `marker`: specifica il tipo di marcatore grafico per i singoli punti (punto, cerchio, triangolo con punte, stella, ecc.) Esso può anche essere personalizzato ulteriormente attraverso l'utilizzo di altri parametri opzionali come `markersize` che ne assegna la dimensione, `markerfacecolor` che ne definisce il colore, ecc.
  - `label`: assegna un'etichetta agli assi (che viene utilizzata nella legenda)
  - `alpha`: specifica la trasparenza (con un valore in singola precisione compreso tra 0 e 1)
- `plt.scatter(X-list, Y-list)`: serve a disegnare un insieme di soli punti sparsi passando separatamente la lista delle ascisse e delle ordinate. Anche per il metodo scatter vi sono alcuni parametri nominali opzionali:
  - `color`: con cui si specifica il colore dei marcatori
  - `marker`: con cui si definisce il tipo di marcatore
  - `linewidth`: con cui si indica lo spessore del bordo del marcatore
  - `edgecolors`: con cui si definisce il colore del bordo del marcatore
- `plt.imshow(X, cmap)`: è un metodo che permette la visualizzazione dei dati come immagini. Come parametri richiede un array o un'immagine. Il parametro `cmap` si riferisce alle cosiddette *colormaps*. Le colormap, o mappe dei colori, in

MatplotLib sono schemi di colori utilizzati per associare i valori di una scala a colori specifici. In altre parole, le colormap sono utilizzate per visualizzare dati numerici tramite una gamma di colori. MatplotLib include diverse colormap predefinite, quali "viridis", "plasma", "inferno", "magma", "cividis", "jet", e molte altre. Esse sostanzialmente sono utilizzate per mappare dati scalari con colori ed ogni colore ha un valore che va da 0 a 1 (in decimale). Le colormap risultano molto utili poiché facilitano la visualizzazione dei dati attraverso varie tecniche grafiche.

- `plt.show()`: è un metodo che visualizza le figure tracciate. Ha l'effetto di lasciare aperta la finestra grafica anche dopo l'esecuzione del programma, ma attraverso il parametro nominale `block`, se gli si assegna il valore `False`, si restituisce il controllo al programma il quale chiude automaticamente la finestra grafica.

- `plt.grid()`: è un metodo che serve a far visualizzare una griglia (incrocio di linee ortogonali) sovrapposta alla figura che aiuta a identificare le posizioni dei vari elementi grafici. Possiede anche alcuni parametri opzionali:

  - `alpha`: con cui si indica la trasparenza (con un valore a precisione singola compreso tra 0 e 1)
  - `color`: per stabilire il colore della linea come stringa
  - `linestyle`: per specificare lo stile della linea
  - `linewidth`: per indicare lo spessore delle linee
  - `axis`: per specificare se visualizzare solo le linee parallele all'asse X ("x"), solo le linee parallele all'asse Y ("y") oppure entrambe ("`both`")

- `plt.title(text)`: imposta uno dei tre titoli disponibili per il grafico, rispettivamente, al centro sopra gli assi, al bordo sinistro o al bordo destro, con i parametri nominali opzionali `loc` che indica la posizione in cui mettere il titolo e può assumere i valori `center`, `left` o `right`.

- `plt.legend()`: genera automaticamente la legenda del grafico facendo riferimento alle etichette delle figure

- `plt.subplot(row, column, index)`: è un metodo che aggiunge un nuovo asse alla figura corrente seguendo uno schema tabulare dove `row` e `column` indicano rispettivamente il numero di righe e di colonne in cui si suddivide idealmente la figura e `index` rappresenta la posizione progressiva in cui tracciare il successivo asse. Ogni volta che si definisce un `subplot`, quello diventa l'asse corrente e le successive operazioni di disegno hanno effetto su di esso.

5.2.3 Il modulo openCV-Python

OpenCV (dall'inglese *Open Source Computer Vision*) è una libreria gratuita per la visualizzazione, l'elaborazione e la manipolazione delle immagini. Essa fornisce una grande varietà di strumenti ed algoritmi che possono essere utilizzati per il riconoscimento delle immagini, la realtà aumentata, la localizzazione di oggetti, la sorveglianza, il conteggio dei veicoli in transito, analisi delle immagini mediche, ecc... OpenCV-Python è una versione della libreria OpenCV (Open Source Computer Vision Library) scritta specificamente per Python. Si tratta di un wrapper Python della libreria OpenCV originale, che è scritta principalmente in C++. Per quanto riguarda l'elaborazione delle immagini, potremmo dire che esso comprende processi di analisi e manipolazione di un'immagine digitale al fine di migliorarne la

qualità. Sostanzialmente vi sono tre passaggi principali: caricamento dell'immagine, analisi e manipolazione della stessa, restituzione dell'immagine alterata. Tra i principali metodi del modulo openCV-Python (che per semplicità verrà richiamato con l'alias "cv") vi sono:

- `cv.imread(path-image, flag)`: è un metodo che legge un'immagine (.jpg, .png, .bmp, ecc...) tramite il suo percorso, mentre il parametro `flag` specifica il modo in cui visualizzare l'immagine. Se il flag non viene specificato, il metodo `cv.imread()` legge le immagini utilizzando lo spazio di colore BGR in cui ogni colore viene definito mediante delle percentuali di blu (Blue), verde (Green) e rosso (Red) esattamente in quest'ordine, che è il formato nativo di MatplotLib sebbene meno diffuso del più comune RGB. In alternativa si possono indicare altre costanti quali:
  - `cv.IMREAD_COLOR`: indica che verrà caricata un'immagine a colori
  - `cv.IMREAD_GRAYSCALE`: indica che verrà caricata un'immagine a toni di grigio
- `cv.imshow(window_name, image)`: è un metodo usato per visualizzare un'immagine. Il parametro `window_name` è una stringa che rappresenta il nome della finestra sulla quale verrà visualizzata l'immagine, mentre `image` è l'immagine da essere visualizzata. Spesso utilizzata con la funzione `waitKey(0)` che serve a mostrare per un certo periodo l'immagine su schermo o, eventualmente, lasciarla finché l'utente non decide di chiuderla e con la funzione `destroyAllWindows()` che serve a cancellare l'immagine dalla memoria.
- `cv.imwrite(filename, image)`: metodo usato per salvare un'immagine che chiede come parametri di ingresso il nome del file e l'immagine da salvare.
- `cv.cvtColor(src, code)`: è un metodo usato per convertire un'immagine da uno spazio di colori ad un altro che chiede in input l'immagine da convertire ed il codice dello spazio dei colori in cui convertirla. Per esempio, per convertire un'immagine da BGR in scala di grigi, si utilizza il seguente parametro: `cv.COLOR_BGR2GRAY`.
- `cv.threshold(src, threshold_value, max_value, type)`: è un metodo che si utilizza per effettuare la cosiddetta sogliatura (dall'inglese *thresholding*), cioè la separazione tra i pixel che superano un certo livello di soglia (i quali diventano bianchi) e quelli che non la superano o sono uguali (i quali diventano neri). In particolare, il primo parametro nominale `src` richiede un'immagine a toni di grigio, il secondo ed il terzo parametro sono il valore base ed il valore soglia, il quarto parametro specifica il tipo di threshold da applicare:
  - `cv.THRESH_BINARY`: se l'intensità del pixel dell'immagine sorgente supera il valore `threshold_value` allora assegna un'intensità pari a `max_value` al pixel della nuova immagine, altrimenti la pone uguale a zero.
  - `cv.THRESH_TRUNC`: se l'intensità del pixel dell'immagine sorgente supera il valore `threshold_value` allora assegna quel valore come intensità del pixel della nuova immagine, altrimenti l'intensità del pixel resta invariata.
  - `cv.THRESH_TOZERO`: se l'intensità del pixel dell'immagine sorgente supera il valore `threshold_value` allora assegna l'intensità del pixel resta inalterata, altrimenti la pone uguale a zero.

5.3 La classe NSmorph

La classe NSmorph è una classe ideata per la gestione di immagini neutrosofiche e l'implementazione dei quattro principali operatori morfologici neutrosofici (dilatazione, erosione, apertura e chiusura neutrosofica) che abbiamo introdotto e studiato nel Capitolo 4. In particolare, il metodo costruttore consente di generare e memorizzare un'immagine neutrosofica a partire da oggetti di tipo diverso. I tipi consentiti dal costruttore possono essere una stringa corrispondente al percorso di un'immagine in formato `png` o `jpg`, una matrice numpy o un altro oggetto della stessa classe NSmorph. Inoltre, le immagini lette devono essere di tipo binario. Il costruttore utilizzato all'interno della suddetta classe è il seguente:

```
import os.path
import cv2 as cv
import numpy as np
from matplotlib import pyplot as plt

class NSmorph:
   def __init__(self, image, raggio=0):
      if raggio < 0:
      raise ValueError(f"Il raggio '{raggio}'dell'intorno non puo'
    essere negativo")
      if type(image) == NSmorph :
         self.__ns_image = image.get()
         self.__image_orig = image.getOrig()
         self.__altezza = image.altezza()
         self.__larghezza = image.larghezza()
         self.__raggio = image.raggio()
      elif type(image) == str:
         tmp_imgns = NSmorph(NSmorph.carica(image), raggio)
         self.__ns_image = tmp_imgns.get()
         self.__image_orig = tmp_imgns.getOrig()
         self.__altezza = tmp_imgns.altezza()
         self.__larghezza = tmp_imgns.larghezza()
         self.__raggio = raggio
      elif type(image) == np.ndarray:
         if image is None:
            raise ValueError("L'immagine non e' valida")
         (altezza, larghezza)=image.shape
         self.__altezza = altezza
         self.__larghezza = larghezza
         self.__image_orig = image
         self.__raggio = raggio
         i_med = np.zeros((altezza, larghezza), dtype = np.float32)
         for y in range(altezza):
            for x in range(larghezza):
               md = 0
               n_pixel = 0
               for j in range(y - raggio, y + raggio + 1):
                  for i in range(x - raggio, x + raggio +1):
                     if (0<=i<larghezza) and (0<=j<altezza) :
                        md += image[j][i]
                        n_pixel += 1
               md /= n_pixel
               i_med[y][x] = md
         i_med_min = i_med.min()
         i_med_max = i_med.max()
         i_med_ampiezza = i_med_max - i_med_min
         delta = np.zeros((altezza, larghezza), dtype=np.float32)
         for y in range(altezza):
            for x in range(larghezza):
               delta[y][x] = abs(image[y][x] - i_med[y][x])
         delta_min = delta.min()
```

```
51          delta_max = delta.max()
52          delta_ampiezza = delta_max - delta_min
53          ns_image = np.zeros((altezza, larghezza, 3), dtype = np.
    float32)
54          for y in range(altezza):
55             for x in range(larghezza):
56                ns_image[y][x][0] = (i_med[y][x] - i_med_min)/
    i_med_ampiezza if i_med_ampiezza !=0 else 0
57                ns_image[y][x][1] = (delta[y][x] - delta_min)/
    delta_ampiezza if delta_ampiezza !=0 else 0
58                ns_image[y][x][2] = 1 - ns_image[y][x][0]
59          self.__ns_image = ns_image
60       else:
61          raise ValueError("Il primo parametro deve essere una matrice
     o una stringa")
```

Nello specifico, dopo aver importato il modulo os.path (il quale consente di effettuare operazioni su file e directory indipendentemente dal sistema operativo utilizzato), il modulo openCV-Python, il modulo Numpy e il modulo pyplot, si crea il costruttore che richiede come parametri un'immagine e il raggio dell'intorno del generico pixel (x, y), che per default è fissato a 0. Innanzitutto si effettua un controllo preventivo sul raggio sollevando, se necessario, un'eccezione qualora il raggio inserito risultasse negativo.

```
1 import os.path
2 import cv2 as cv
3 import numpy as np
4 from matplotlib import pyplot as plt

6 class NSmorph:
7    def __init__(self, image, raggio=0):
8       if raggio < 0:
9       raise ValueError(f"Il raggio '{raggio}'dell'intorno non puo'
    essere negativo")
```

Innanzitutto, vengono effettuati i controlli sul tipo di parametro passato come immagine. In particolare se il parametro è un oggetto immagine neutrosofica, allora vengono memorizzate le proprietà corrispondenti:

```
1 if type(image) == NSmorph :
2    self.__ns_image = image.get()
3    self.__image_orig = image.getOrig()
4    self.__altezza = image.altezza()
5    self.__larghezza = image.larghezza()
6    self.__raggio = image.raggio()
```

Se il parametro passato è di tipo stringa, l'oggetto viene creato a partire dal caricamento di un file immagine dal disco ed attraverso l'utilizzo del metodo statico `carica(img_path)` e le sue proprietà vengono memorizzate come segue:

```
1  elif type(image) == str:
2          tmp_imgns = NSmorph(NSmorph.carica(image), raggio)
3    self.__ns_image = tmp_imgns.get()
4    self.__image_orig = tmp_imgns.getOrig()
5    self.__altezza = tmp_imgns.altezza()
```

```
6     self.__larghezza = tmp_imgns.larghezza()
7     self.__raggio = raggio
```

Se il parametro passato è di tipo ndarray, dopo un controllo sul parametro stesso, si crea un oggetto partendo da una matrice numpy di valori ed inoltre si assegna ad una tupla le dimensioni dell'immagine di partenza memorizzandole successivamente come proprietà interne alla classe.

```
1  elif type(image) == np.ndarray:
2     if image is None:
3        raise ValueError("L'immagine non e' valida")
4     (altezza, larghezza)=image.shape
5     self.__altezza = altezza
6     self.__larghezza = larghezza
7     self.__image_orig = image
8     self.__raggio = raggio
9     i_med = np.zeros((altezza, larghezza), dtype = np.float32)
10    for y in range(altezza):
11       for x in range(larghezza):
12          md = 0
13          n_pixel = 0
14          for j in range(y - raggio, y + raggio + 1):
15             for i in range(x - raggio, x + raggio +1):
16                if (0<=i<larghezza) and (0<=j<altezza) :
17                   md += image[j][i]
18                   n_pixel += 1
19          md /= n_pixel
20          i_med[y][x] = md
21    i_med_min = i_med.min()
22    i_med_max = i_med.max()
23    i_med_ampiezza = i_med_max - i_med_min
24    delta = np.zeros((altezza, larghezza), dtype=np.float32)
25    for y in range(altezza):
26       for x in range(larghezza):
27          delta[y][x] = abs(image[y][x] - i_med[y][x])
28    delta_min = delta.min()
29    delta_max = delta.max()
30    delta_ampiezza = delta_max - delta_min
31    ns_image = np.zeros((altezza, larghezza, 3), dtype = np.float32)
32    for y in range(altezza):
33       for x in range(larghezza):
34          ns_image[y][x][0] = (i_med[y][x] - i_med_min)/i_med_ampiezza
        if i_med_ampiezza !=0 else 0
35          ns_image[y][x][1] = (delta[y][x] - delta_min)/delta_ampiezza
        if delta_ampiezza !=0 else 0
36          ns_image[y][x][2] = 1 - ns_image[y][x][0]
37    self.__ns_image = ns_image
38 else:
39    raise ValueError("Il primo parametro deve essere una matrice o una
        stringa")
```

Si noti (dalla riga 10 alla riga 19) come nel calcolo dei valori della funzione $\overline{I}$di intensità media, a differenza della definizione formale fornita nella formula 4.1, il valore dell'intensità media di ciascun pixel non viene calcolata dividendo per il quadrato dell'ampiezza $(\omega + 1)^2$, ma per il numero effettivo di pixel (qui rappresentato dalla variabile `n_pixel`) coperti dall'elemento strutturante centrato nel pixel di riferimento rispetto all'immagine principale, essendo evidente che, quando durante la scansione il pixel si avvicina ai bordi dell'immagine, l'area d'intersezione di questa con il kernel si riduce notevolmente.

Subito dopo le assegnazioni, si crea una matrice di zeri richiamando il metodo `np.zeros()` avente come tupla di ingresso, una tupla avente come elementi le dimensioni dell'immagine precedentemente assegnata e ponendo `dtype = np.float32`.

```
i_med = np.zeros((altezza, larghezza), dtype = np.float32)
for y in range(altezza):
   for x in range(larghezza):
      md = 0
      n_pixel = 0
      for j in range(y - raggio, y + raggio + 1):
         for i in range(x - raggio, x + raggio +1):
            if (0<=i<larghezza) and (0<=j<altezza) :
               md += image[j][i]
               n_pixel += 1
      md /= n_pixel
      i_med[y][x] = md/n_pixel
```

Per costruire l'immagine neutrosofica è necessario creare la tripla $(\mu_I, \sigma_I, \omega_I)$ associata all'intensità del pixel $I(x, y)$. A tale scopo si utilizzano la definizione della funzione intensità media $\overline{I}$ ed il valore di picco massimo $\overline{I}_{max}$ e minimo $\overline{I}_{min}$e la definizione di funzione di omogeneità $\delta(x, y)$ ed il valore di picco massimo δ_{max} e minimo δ_{min}.

```

        i_med_min = i_med.min()
        i_med_max = i_med.max()
        i_med_ampiezza = i_med_max - i_med_min
        delta = np.zeros((altezza, larghezza), dtype=np.float32)
        for y in range(altezza):
            for x in range(larghezza):
                delta[y][x] = abs(image[y][x] - i_med[y][x])
        delta_min = delta.min()
        delta_max = delta.max()
        delta_ampiezza = delta_max - delta_min
```

Dopo aver introdotto le funzioni necessarie, la definizione dell'immagine neutrosofica si ottiene calcolando le tre funzioni della tripla $(\mu_I, \sigma_I, \omega_I)$.

```
ns_image = np.zeros((altezza, larghezza, 3), dtype = np.float32)
for y in range(altezza):
   for x in range(larghezza):
      ns_image[y][x][0] = (i_med[y][x] - i_med_min)/i_med_ampiezza
    if i_med_ampiezza !=0 else 0
      ns_image[y][x][1] = (delta[y][x] - delta_min)/delta_ampiezza if
     delta_ampiezza !=0 else 0
      ns_image[y][x][2] = 1 - ns_image[y][x][0]
self.__ns_image = ns_image
```

Nel caso in cui il parametro passato non è né un'immagine neutrosofica, né una stringa, né una matrice, viene sollevato un *ValueError*.

```
else:
   raise ValueError("Il primo parametro deve essere una matrice o una
     stringa")
```

La classe NSmorph dispone di alcuni metodi che permettono la restituzione dell'immagine neutrosofica nella sua interezza o dell'immagine binaria dalla quale proviene; altri metodi che restituiscono o assegnano una tra le funzioni di appartenenza, indeterminazione e non appartenenza.
Il metodo statico `carica(img_path)` restituisce un'immagine in toni di grigio creandola direttamente dal percorso dell'immagine `img_path` e sollevando errori nel caso in cui il percorso dell'immagine risulti invalido, non trovato (mediante il metodo `os.path.isfile()`) o non accessibile.

```
@staticmethod
def carica(img_path):
    if img_path is None:
        raise ValueError("Il percorso dell'immagine non e' valido")
    if not os.path.isfile(img_path):
        raise FileNotFoundError(f"Il file '{img_path}' non esiste o non e' accessibile.")
    image = cv.imread(img_path, cv.IMREAD_GRAYSCALE)
    if image is None:
        raise IOError(f"Non e' stato possibile leggere il file immagine '{img_path}'")
    return image
```

Il metodo `get()` restituisce l'immagine neutrosofica.

```
def get(self):
    return self.__ns_image
```

I metodi `getM()`, `getI()` e `getNM()` restituiscono rispettivamente il livello dell'immagine neutrosofica in termini dei gradi di appartenenza, di indeterminazione e di non appartenenza.

```
def getM(self):
    return self.__ns_image[:,:,0]

def getI(self):
    return self.__ns.image[:,:,1]

def getNM(self):
    return self.__ns.image[:,:,2]
```

I metodi `setM()`, `setI()`, `setNM()` assegnano rispettivamente il livello dell'immagine neutrosofica in termini dei loro gradi di appartenenza, indeterminazione e non appartenenza

```
def setM(self, x, y, mu):
    self.__ns_image[y][x][0] = mu

def setI(self, x, y, sigma):
    self.__ns_image[y][x][1] = sigma

def setNM(self, x, y, omega):
    self.__ns_image[y][x][2] = omega
```

Il metodo `getOrig()` restituisce l'immagine originale.

```
def getOrig(self):
    return self.__image
```

I metodi `larghezza()` ed `altezza()` restituiscono rispettivamente la larghezza e l'altezza dell'immagine e saranno utili in seguito nella costruzione dei metodi di dilatazione e di erosione.

```
def larghezza(self):
    return self.__larghezza

def altezza(self):
    return self.__altezza
```

Il metodo `raggio()` restituisce il raggio dell'intorno associato all'immagine caricata.

```
def raggio(self):
    return self.__raggio
```

Il metodo `getBinaria()` applica il thresholding all'immagine corrente rispetto ad un valore soglia passato come parametro dall'utente, restituisce l'immagine binaria come matrice numpy di righe e colonne con origine $(0, 0)$ in alto a sinistra ed avente valori 0=nero e 1=bianco.

```
def getBinaria(self, soglia):
    (ret, bin_image) = cv.threshold(self.__image_orig, soglia, 1, cv.
    THRESH_BINARY)
    return bin_image
```

Il metodo `getRappresentazione()` restituisce un'immagine a toni di grigio interpolando i tre livelli presi con dei pesi ed un valore soglia di default.

```
def getRappresentazione(self, pesoM=0.85, pesoI=0.25, pesoNM=-0.1,
    binaria =False, soglia=128):
    img_M = self.getM()
    img_I = self.getI()
    img_NM = self.getNM()
    mat_rap = np.zeros((self.__altezza, self.__larghezza, 3), dtype = np
    .float32)
    for y in range(self.__altezza):
        for x in range(self.__larghezza):
            mat_rap[y][x][0] = pesoM*img_M[y][x] + pesoI*img_I[y][x] +
    pesoNM*img_NM[y][x]
    img_rap = cv.cvtColor(mat_rap, cv.COLOR_BGR2GRAY)
    if binaria == True:
        (ret, img_rap) = cv.threshold(img_rap, soglia, 1, cv.
    THRESH_BINARY)
    return img_rap
```

La classe NSmorph dispone di altri metodi che permettono la manipolazione dell'immagine stessa attraverso un'altra immagine. Più precisamente, un'immagine neutrosofica può essere manipolata attraverso l'utilizzo degli operatori di dilatazione, erosione, apertura e chiusura neutrosofica definiti precedentemente.
Il metodo `dilatazione(self, kernel)` restituisce l'immagine neutrosofica dilatata mediante un elemento strutturante (o *kernel*), anch'esso immagine neutrosofica, passato come parametro (vedi Def. 4.2.3).

```
def dilatazione(self, kernel):
    (altezza, larghezza) = (self.__altezza, self.__larghezza)
    (altezza_k, larghezza_k) = (kernel.altezza(), kernel.larghezza())

    img_M = self.getM()
    img_I = self.getI()
    img_NM = self.getNM()
    kernel_M = kernel.getM()
    kernel_I = kernel.getI()
    kernel_NM = kernel.getNM()

    mat_generatrice = np.zeros((altezza, larghezza, 3), dtype=np.uint8
    )
    im_vuota = cv.cvtColor(mat_generatrice, cv.COLOR_BGR2GRAY)
    im_dil = NSmorph(im_vuota)

    for y in range(altezza):
        for x in range(larghezza):
            mat_M = img_M[y: y + altezza_k, x: x + larghezza_k]
            mat_I = img_I[y:y + altezza_k, x:x + larghezza_k]
            mat_NM = img_NM[y:y + altezza_k, x:x + larghezza_k]

            (n_righe, n_colonne) = mat_M.shape
            mat_kernelM = kernel_M[0:n_righe, 0:n_colonne]
            mat_kernelI = kernel_I[0:n_righe, 0:n_colonne]
            mat_kernelNM = kernel_NM[0:n_righe, 0:n_colonne]

            minimi_M = np.minimum(mat_M, mat_kernelM)
            minimi_I = np.minimum(mat_I, mat_kernelI)
            massimi_NM = np.maximum(mat_NM, 1 - mat_kernelNM)

            mu = minimi_M.max()
            sigma = minimi_I.max()
            omega = massimi_NM.min()

            im_dil.setM(x, y, mu)
            im_dil.setI(x, y, sigma)
            im_dil.setNM(x, y, omega)
    return im_dil
```

Nelle prime righe, il metodo salva su due tuple diverse le dimensioni dell'immagine neutrosofica e dell'elemento strutturante ed assegna rispettivamente a `img_M`, `img_I` , `img_NM` i livelli di appartenenza, indeterminazione e non appartenenza dell'immagine neutrosofica alla quale vogliamo applicare la dilatazione.

```
def dilatazione(self, kernel):
    (altezza, larghezza) = (self.__altezza, self.__larghezza)
    (altezza_k, larghezza_k) = (kernel.altezza(), kernel.larghezza())

    img_M = self.getM()
    img_I = self.getI()
```

```
    img_NM = self.getNM()
    kernel_M = kernel.getM()
    kernel_I = kernel.getI()
    kernel_NM = kernel.getNM()
```

Inoltre viene generata la matrice `mat_generatrice` con il metodo `np.zeros()` dalla libreria NumPy la quale serve a costruire l'immagine neutrosofica `im_dil` a partire dall'immagine vuota `im_vuota` creata utilizzando il metodo `cv.cvtColor()` della libreria OpenCV-Python e convertita in toni di grigi utilizzando la costante `cv.COLOR_BGR2GRAY`.

```
mat_generatrice = np.zeros((altezza, larghezza, 3), dtype=np.uint8)
im_vuota = cv.cvtColor(mat_generatrice, cv.COLOR_BGR2GRAY)
im_dil = NSmorph(im_vuota)
```

Dopo aver generato la matrice, si scorrono i singoli pixel dell'immagine di coordinate (x, y) per poter effettuare tutte le operazioni coinvolte nel processo di dilatazione dell'immagine. In questi due cicli i valori di y e x variano rispettivamente all'interno di `range(altezza)` e `range(larghezza)`. All'interno di essi, invece, vi è la vera e propria operazione di dilatazione effettuata su ogni pixel.

```
for y in range(altezza):
    for x in range(larghezza):
        mat_M = img_M[y: y + altezza_k, x: x + larghezza_k]
        mat_I = img_I[y:y + altezza_k, x:x + larghezza_k]
        mat_NM = img_NM[y:y + altezza_k, x:x + larghezza_k]

        (n_righe, n_colonne) = mat_M.shape
        mat_kernelM = kernel_M[0:n_righe, 0:n_colonne]
        mat_kernelI = kernel_I[0:n_righe, 0:n_colonne]
        mat_kernelNM = kernel_NM[0:n_righe, 0:n_colonne]

        minimi_M = np.minimum(mat_M, mat_kernelM)
        minimi_I = np.minimum(mat_I, mat_kernelI)
        massimi_NM = np.maximum(mat_NM, 1 - mat_kernelNM)

        mu = minimi_M.max()
        sigma = minimi_I.max()
        omega = massimi_NM.min()

        im_dil.setM(x, y, mu)
        im_dil.setI(x, y, sigma)
        im_dil.setNM(x, y, omega)
```

In particolare, si estraggono le matrici di appartenenza, indeterminazione e non appartenenza della stessa dimensione dell'elemento strutturante a partire dalla posizione (x, y).

```
mat_M = img_M[y: y + altezza_k, x: x + larghezza_k]
mat_I = img_I[y:y + altezza_k, x:x + larghezza_k]
mat_NM = img_NM[y:y + altezza_k, x:x + larghezza_k]
```

Si calcolano, inoltre, le dimensioni effettive delle matrici (che risultano essere tutte uguali) nel caso in cui si arrivi vicino ai bordi (destro ed inferiore).

```
(n_righe, n_colonne) = mat_M.shape
```

Si riducono le dimensioni dell'elemento strutturante alle dimensioni della matrice estratta.

```
mat_kernelM = kernel_M[0:n_righe, 0:n_colonne]
mat_kernelI = kernel_I[0:n_righe, 0:n_colonne]
mat_kernelNM = kernel_NM[0:n_righe, 0:n_colonne]
```

Si calcolano i minimi (per mat_M e mat_I) o i massimi (per mat_NM) degli elementi corrispondenti delle matrici estratte e degli elementi strutturanti (o di 1 meno l'elemento strutturante nel caso di mat_M).

```
minimi_M = np.minimum(mat_M, mat_kernelM)
minimi_I = np.minimum(mat_I, mat_kernelI)
massimi_NM = np.maximum(mat_NM, 1 - mat_kernelNM)
```

Si calcolano i valori dei gradi di appartenenza, indeterminazione e non appartenenza dei pixel di coordinate (x, y) come il massimo o il minimo delle matrici ottenute.

```
mu = minimi_M.max()
sigma = minimi_I.max()
omega = massimi_NM.min()
```

Si memorizzano i valori dei tre gradi di appartenenza nell'immagine `im_dil` in corrispondenza del pixel di coordinate (x, y) con i metodi `setM()`, `setI()` e `setNM()`. Infine, viene restituita l'immagine neutrosofica dilatata.

```
im_dil.setM(x, y, mu)
im_dil.setI(x, y, sigma)
im_dil.setNM(x, y, omega)
return im_dil
```

Il metodo `erosione(self, kernel)` restituisce l'immagine neutrosofica erosa mediante un elemento strutturante (o *kernel*), anch'esso immagine neutrosofica, passato come parametro (vedi Def. 4.2.8).

```
def erosione(self, kernel):
    (altezza, larghezza) = (self.__altezza, self.__larghezza)
    (altezza_k, larghezza_k) = (kernel.altezza(), kernel.larghezza())

    img_M = self.getM()
    img_I = self.getI()
    img_NM = self.getNM()
    kernel_M = kernel.getM()
    kernel_I = kernel.getI()
    kernel_NM = kernel.getNM()
```

```
11
12      mat_generatrice = np.zeros((altezza, larghezza, 3), dtype=np.uint8
        )
13      im_vuota = cv.cvtColor(mat_generatrice, cv.COLOR_BGR2GRAY)
14      im_er = NSmorph(im_vuota)
15
16      for y in range(altezza):
17          for x in range(larghezza):
18              mat_M = img_M[y: y + altezza_k, x: x + larghezza_k]
19              mat_I = img_I[y:y + altezza_k, x:x + larghezza_k]
20              mat_NM = img_NM[y:y + altezza_k, x:x + larghezza_k]
21
22              (n_righe, n_colonne) = mat_M.shape
23              mat_kernelM = kernel_M[0:n_righe, 0:n_colonne]
24              mat_kernelI = kernel_I[0:n_righe, 0:n_colonne]
25              mat_kernelNM = kernel_NM[0:n_righe, 0:n_colonne]
26
27              massimi_M = np.maximum(mat_M, 1 - mat_kernelM)
28              massimi_I = np.maximum(mat_I, 1 - mat_kernelI)
29              minimi_NM = np.minimum(mat_NM, mat_kernelNM)
30
31              mu = massimi_M.min()
32              sigma = massimi_I.min()
33              omega = minimi_NM.max()
34
35              im_er.setM(x, y, mu)
36              im_er.setI(x, y, sigma)
37              im_er.setNM(x, y, omega)
38      return im_er
```

Analogamente al metodo di dilatazione neutrosofica, anche nel caso del metodo di erosione si ha la memorizzazione delle dimensioni dell'immagine neutrosofica e dell'elemento strutturante su due tuple diverse. Vi è inoltre l'assegnazione dei livelli di appartenenza, indeterminazione e non appartenenza dell'immagine neutrosofica sulla quale vogliamo applicare l'erosione alle variabili `img_M`, `img_I`, `img_NM`.

```
1  def erosione(self, kernel):
2      (altezza, larghezza) = (self.__altezza, self.__larghezza)
3      (altezza_k, larghezza_k) = (kernel.altezza(), kernel.larghezza())
4
5      img_M = self.getM()
6      img_I = self.getI()
7      img_NM = self.getNM()
8      kernel_M = kernel.getM()
9      kernel_I = kernel.getI()
10     kernel_NM = kernel.getNM()
```

Dopo aver effettuato l'assegnazione, viene generata la matrice `mat_generatrice` con il metodo `np.zeros()` dalla libreria NumPy la quale serve a costruire l'immagine neutrosofica `im_dil` a partire dall'immagine vuota `im_vuota` creata utilizzando il metodo `cv.cvtColor()` della libreria OpenCV-Python e convertita in toni di grigi utilizzando il parametro nominale `cv.COLOR_BGR2GRAY`.

```
1  mat_generatrice = np.zeros((altezza, larghezza, 3), dtype=np.uint8)
2  im_vuota = cv.cvtColor(mat_generatrice, cv.COLOR_BGR2GRAY)
3  im_er = NSmorph(im_vuota)
```

Successivamente si scorrono i singoli pixel dell'immagine di coordinate (x, y) per poter effettuare tutte le operazioni coinvolte nel processo di erosione dell'immagine. I due cicli **`for`** scorrono tutti i pixel. In questi due cicli i valori di y e x variano rispettivamente all'interno di `range(altezza)` e `range(larghezza)`.

```
for y in range(altezza):
   for x in range(larghezza):
      mat_M = img_M[y: y + altezza_k, x: x + larghezza_k]
      mat_I = img_I[y:y + altezza_k, x:x + larghezza_k]
      mat_NM = img_NM[y:y + altezza_k, x:x + larghezza_k]

      (n_righe, n_colonne) = mat_M.shape
      mat_kernelM = kernel_M[0:n_righe, 0:n_colonne]
      mat_kernelI = kernel_I[0:n_righe, 0:n_colonne]
      mat_kernelNM = kernel_NM[0:n_righe, 0:n_colonne]

      massimi_M = np.maximum(mat_M, 1 - mat_kernelM)
      massimi_I = np.maximum(mat_I, 1 - mat_kernelI)
      minimi_NM = np.minimum(mat_NM, mat_kernelNM)

      mu = massimi_M.min()
      sigma = massimi_I.min()
      omega = minimi_NM.max()

      im_er.setM(x, y, mu)
      im_er.setI(x, y, sigma)
      im_er.setNM(x, y, omega)
```

In particolare, si estraggono le matrici di appartenenza, indeterminazione e non appartenenza della stessa dimensione dell'elemento strutturante a partire dalla posizione (x, y).

```
mat_M = img_M[y: y + altezza_k, x: x + larghezza_k]
mat_I = img_I[y:y + altezza_k, x:x + larghezza_k]
mat_NM = img_NM[y:y + altezza_k, x:x + larghezza_k]
```

Si calcolano, inoltre, le dimensioni effettive delle tre matrici corrispondenti ai valori di appartenenza, indeteminazione e non appartenenza (che risultano essere tutte uguali) nel caso in cui si arrivi vicino ai bordi (destro ed inferiore).

```
(n_righe, n_colonne) = mat_M.shape
```

Si riducono le componenti dell'elemento strutturante alle dimensioni della matrice estratta.

```
mat_kernelM = kernel_M[0:n_righe, 0:n_colonne]
mat_kernelI = kernel_I[0:n_righe, 0:n_colonne]
mat_kernelNM = kernel_NM[0:n_righe, 0:n_colonne]
```

Si calcolano i massimi (per mat_M e mat_I) o i minimi (per mat_NM) degli elementi corrispondenti delle matrici estratte e degli elementi strutturanti (o di 1 meno l'elemento strutturante nel caso di mat_M e mat_I).

```
massimi_M = np.maximum(mat_M, 1 - mat_kernelM)
massimi_I = np.maximum(mat_I, 1 - mat_kernelI)
minimi_NM = np.minimum(mat_NM, mat_kernelNM)
```

Inoltre si calcolano i valori dei gradi di appartenenza, indeterminazione e non appartenenza dei pixel di coordinate (x, y) come massimo o il minimo delle matrici ottenute.

```
mu = massimi_M.min()
sigma = massimi_I.min()
omega = minimi_NM.max()
```

Si memorizzano i valori dei tre gradi di appartenenza nell'immagine `im_er` in corrispondenza del pixel di coordinate (x, y) attraverso i metodi `setM()`, `setI()` e `setNM()`. Infine, viene restituita l'immagine neutrosofica erosa.

```
im_er.setM(x, y, mu)
im_er.setI(x, y, sigma)
im_er.setNM(x, y, omega)
return im_er
```

Il metodo `apertura(self, kernel)` restituisce l'apertura di un'immagine neutrosofica ottenuta applicando all'immagine stessa un'erosione e successivamente una dilatazione mediante lo stesso elemento strutturante (o *kernel*), anch'esso immagine neutrosofica, passato come parametro (si veda Def. 4.2.12).

```
def apertura(self, kernel):
    return self.erosione(kernel).dilatazione(kernel)
```

Il metodo `chiusura(self, kernel)` restituisce la chiusura di un'immagine neutrosofica ottenuta applicando all'immagine stessa una dilatazione e successivamente un'erosione mediante lo stesso elemento strutturante (o *kernel*), anch'esso immagine neutrosofica, passato come parametro (si veda Def. 4.2.15).

```
def chiusura(self, kernel):
    return self.dilatazione(kernel).erosione(kernel)
```

5.4 Esempi di utilizzo della classe NSmorph

Illustriamo adesso l'utilizzo e le potenzialità della classe appena descritta, mediante la scrittura di un breve codice che la richiama, utilizzando in particolare i metodi di dilatazione ed erosione neutrosofica.

```
from NSmorph import NSmorph
import cv2 as cv
import numpy as np
from matplotlib import pyplot as plt

path_immagine = "immagini/j_pepper_noise.jpg"
immagine_jpg = NSmorph.carica(path_immagine)
immagine = NSmorph(immagine_jpg, 4)

path_nucleo = "immagini/kernel_croce.jpg"
nucleo_jpg = NSmorph.carica(path_nucleo)
nucleo = NSmorph(nucleo_jpg, 1)

im_dil = immagine.dilatazione(nucleo)

plt.suptitle("Dilatazione neutrosofica (M=appartenenza, I=
    indeterminazione, NM=non appartenenza)")

plt.subplot(4,3,1)
plt.imshow(immagine.getOrig(), cmap='gray')
plt.xticks([])
plt.yticks([])
plt.title("originale")

plt.subplot(12,9,5)
plt.imshow(nucleo.getOrig(), cmap='gray')
plt.xticks([])
plt.yticks([])
plt.title("kernel")

plt.subplot(12,9,22)
plt.imshow(nucleo.getM(), cmap='gray')
plt.xticks([])
plt.yticks([])
plt.title("kernel M")

plt.subplot(12,9,23)
plt.imshow(nucleo.getI(), cmap='gray')
plt.xticks([])
plt.yticks([])
plt.title("kernel I")

plt.subplot(12,9,24)
plt.imshow(nucleo.getNM(), cmap='gray')
plt.xticks([])
plt.yticks([])
plt.title("kernel NM")

plt.subplot(4,3,3)
plt.imshow(immagine.getRappresentazione(), cmap='gray')
plt.xticks([])
plt.yticks([])
plt.title("neutrosofica")

plt.subplot(4,3,4)
plt.imshow(immagine.getM(), cmap='gray')
plt.xticks([])
plt.yticks([])
plt.title("M")

plt.subplot(4,3,5)
plt.imshow(immagine.getI(), cmap='gray')
plt.xticks([])
plt.yticks([])
plt.title("I")
```

```
plt.subplot(4,3,6)
plt.imshow(immagine.getNM(), cmap='gray')
plt.xticks([])
plt.yticks([])
plt.title("NM")

plt.subplot(4,3,7)
plt.imshow(im_dil.getM(), cmap='gray')
plt.xticks([])
plt.yticks([])
plt.title("dilatazione M")

plt.subplot(4,3,8)
plt.imshow(im_dil.getI(), cmap='gray')
plt.xticks([])
plt.yticks([])
plt.title("dilatazione I")

plt.subplot(4,3,9)
plt.imshow(im_dil.getNM(), cmap='gray')
plt.xticks([])
plt.yticks([])
plt.title("dilatazione NM")

plt.subplot(4,3,11)
plt.imshow(im_dil.getRappresentazione(), cmap='gray')
plt.xticks([])
plt.yticks([])
plt.title("dilatazione")


plt.show()
```

Innanzitutto, si caricano i percorsi dell'immagine e dell'elemento strutturante scegliendo rispettivamente come raggio dell'intorno associato all'immagine neutrosofica un raggio pari a 4 e per l'elemento strutturante pari a 1. Successivamente si crea l'immagine neutrosofica dilatata `im_dil` mediante l'elemento strutturante (anch'esso immagine neutrosofica creata a partire dallo stesso costruttore della classe).

```
path_immagine = "immagini/j_pepper_noise.jpg"
immagine_jpg = NSmorph.carica(path_immagine)
immagine = NSmorph(immagine_jpg, 4)

path_nucleo = "immagini/kernel_croce.jpg"
nucleo_jpg = NSmorph.carica(path_nucleo)
nucleo = NSmorph(nucleo_jpg, 1)

im_dil = immagine.dilatazione(nucleo)
```

Dopo aver caricato i percorsi, si generano le immagini tramite il metodo `plt.subplot` ed in particolare vengono visualizzate a schermo l'immagine originale, il kernel ed i suoi tre livelli di appartenenza (M), indeterminazione (I), non appartenenza (NM) rappresentati come immagini a livelli di grigio, nonché l'immagine neutrosofica interpolata rappresentata anch'essa come immagine a toni di grigio.

```
plt.suptitle("Dilatazione neutrosofica (M=appartenenza, I=
    indeterminazione, NM=non appartenenza)")
```

```
plt.subplot(4,3,1)
plt.imshow(immagine.getOrig(), cmap='gray')
plt.xticks([])
plt.yticks([])
plt.title("originale")

plt.subplot(12,9,5)
plt.imshow(nucleo.getOrig(), cmap='gray')
plt.xticks([])
plt.yticks([])
plt.title("kernel")

plt.subplot(12,9,22)
plt.imshow(nucleo.getM(), cmap='gray')
plt.xticks([])
plt.yticks([])
plt.title("kernel M")

plt.subplot(12,9,23)
plt.imshow(nucleo.getI(), cmap='gray')
plt.xticks([])
plt.yticks([])
plt.title("kernel I")

plt.subplot(12,9,24)
plt.imshow(nucleo.getNM(), cmap='gray')
plt.xticks([])
plt.yticks([])
plt.title("kernel NM")

plt.subplot(4,3,3)
plt.imshow(immagine.getRappresentazione(), cmap='gray')
plt.xticks([])
plt.yticks([])
plt.title("neutrosofica")
```

Nella stessa finestra grafica vengono visualizzati anche i tre livelli di appartenenza dell'immagine neutrosofica, i tre livelli di appartenenza dell'immagine neutrosofica dilatata e l'immagine neutrosofica dilatata.

```
plt.subplot(4,3,4)
plt.imshow(immagine.getM(), cmap='gray')
plt.xticks([])
plt.yticks([])
plt.title("M")

plt.subplot(4,3,5)
plt.imshow(immagine.getI(), cmap='gray')
plt.xticks([])
plt.yticks([])
plt.title("I")

plt.subplot(4,3,6)
plt.imshow(immagine.getNM(), cmap='gray')
plt.xticks([])
plt.yticks([])
plt.title("NM")

plt.subplot(4,3,7)
plt.imshow(im_dil.getM(), cmap='gray')
plt.xticks([])
```

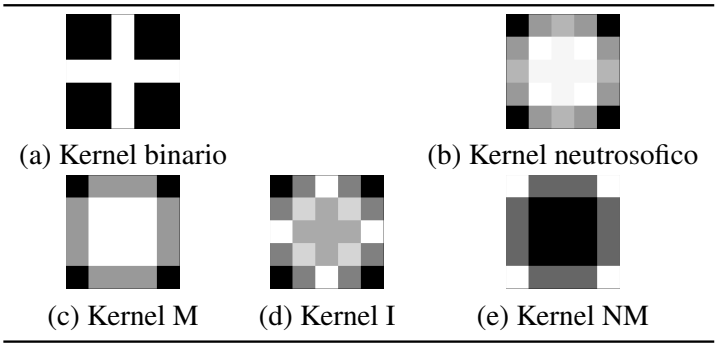
```
plt.yticks([])
plt.title("dilatazione M")

plt.subplot(4,3,8)
plt.imshow(im_dil.getI(), cmap='gray')
plt.xticks([])
plt.yticks([])
plt.title("dilatazione I")

plt.subplot(4,3,9)
plt.imshow(im_dil.getNM(), cmap='gray')
plt.xticks([])
plt.yticks([])
plt.title("dilatazione NM")

plt.subplot(4,3,11)
plt.imshow(im_dil.getRappresentazione(), cmap='gray')
plt.xticks([])
plt.yticks([])
plt.title("dilatazione")
```

Possiamo applicare il programma Python appena descritto per valutare la dilatazione, l'erosione, l'apertura e la chiusura neutrosofica di due semplici immagini binarie di test costituite da una lettera "j" in corsivo che presentano un rumore, ossia una variazione indesiderata e casuale di alcun pixel che ne compromette la chiarezza, la nitidezza e la fedeltà dei dettagli e precisamente:

- un rumore di tipo pepe (pepper noise) che è caratterizzato dalla presenza di pixel neri sparsi casualmente nell'immagine
- un rumore di tipo sale (salt noise) caratterizzato, invece, dalla presenza casuale di pixel bianchi.

Più precisamente, esamineremo il comportamento dell'immagine della j col rumore di tipo pepe rispetto agli operatori di dilatazione (Figura 5.2) e chiusura neutrosofica (Figura 5.5) e dell'immagine della j col rumore di tipo sale rispetto agli operatori di erosione (Figura 5.3) e apertura neutrosofica (Figura 5.4), rispetto ad un elemento strutturante costituito da un kernel a croce di 5×5 pixel trasformato in una immagine neutrosofica con raggio di valore 1 (si veda la Figura 5.1).

Figura 5.1: Kernel croce 5×5 e raggio $r = 1$



(a) Kernel binario (b) Kernel neutrosofico

(c) Kernel M (d) Kernel I (e) Kernel NM

In ciascuno di questi casi, a partire dall'immagine binaria originale, otterremo l'immagine neutrosofica interpolata (secondo la procedura illustrata nella sezione 4.1) e la rappresenteremo per semplicità come immagine a livelli di grigio nonché le sue tre componenti

neutrosofiche di appartenenza (membership, indicata con M), indeterminazione (indeterminacy, indicata con I) e non appartenenza (non membership, indicata con NM), rappresentate anch'esse come immagini a livelli di grigio. Nelle Figure 5.2, 5.3, 5.4 e 5.5 sono poi rappresentate rispettivamente la dilatazione, l'erosione, l'apertura e la chiusura neutrosofica di tali immagini nelle loro tre componenti neutrosofiche di appartenenza (M), indeterminazione (I) e non appartenenza (NM) come immagini a toni di grigio, nonché la corrispondente immagine neutrosofica interpolata (rappresentata a livelli di grigi) e la sua binarizzazione ottenuta con una semplice operazione di sogliatura (thresholding).

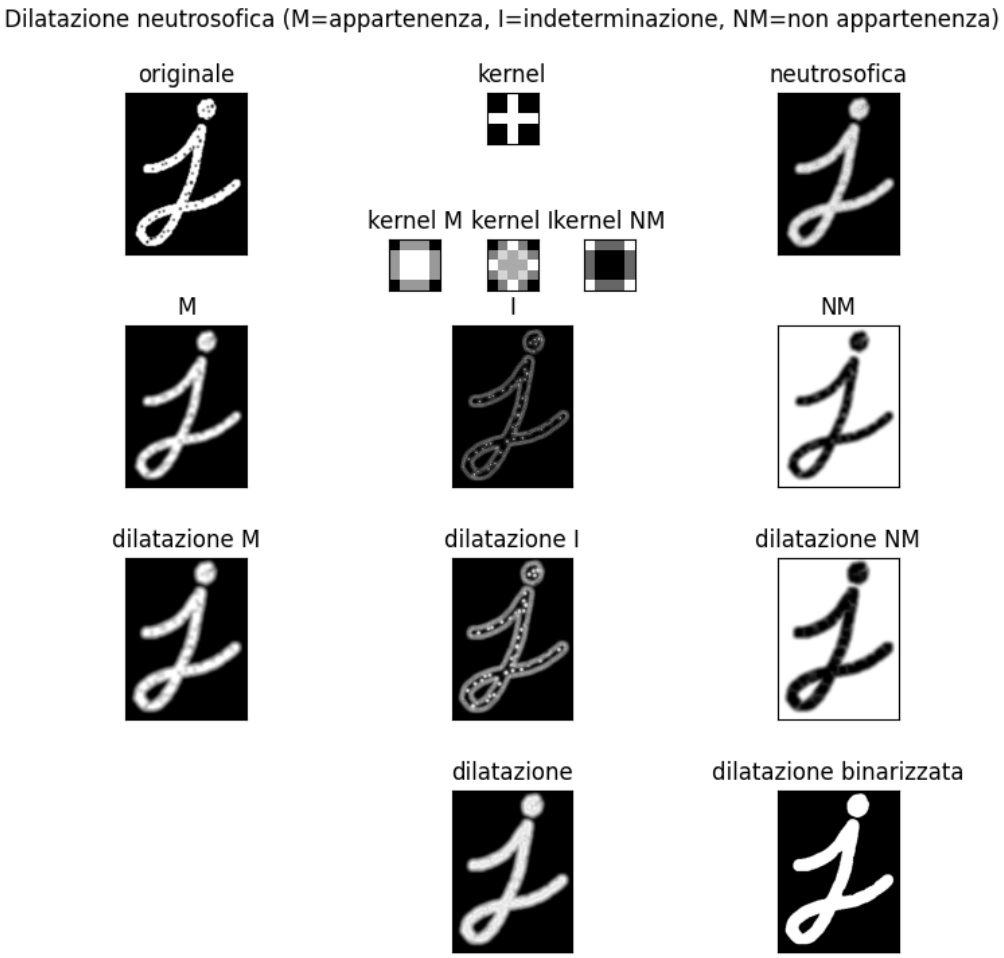


Figura 5.2: Immagine neutrosofica di una j in corsivo con rumore di tipo pepe e sua dilatazione con raggio immagine = 4 e raggio kernel =1

Già ad una prima sommaria osservazione della Figura 5.2, è possibile notare che l'operatore di dilatazione neutrosofica applicato all'immagine col rumore mantiene la traccia del rumore stesso (sia esso di tipo sale che di tipo pepe) che è visibile maggiormente nella componente I di indeterminazione, pur non alterando l'azione tipica della dilatazione, ossia quella di smussamento del profilo dell'immagine e di riempimento delle irregolarità.

Da una prima osservazione della Figura 5.3, è possibile apprezzare che anche l'operatore di erosione neutrosofica applicato all'immagine col rumore mantiene la traccia del rumore stesso. Esso è maggiormente visibile nelle componenti M di appartenenza ed NM di

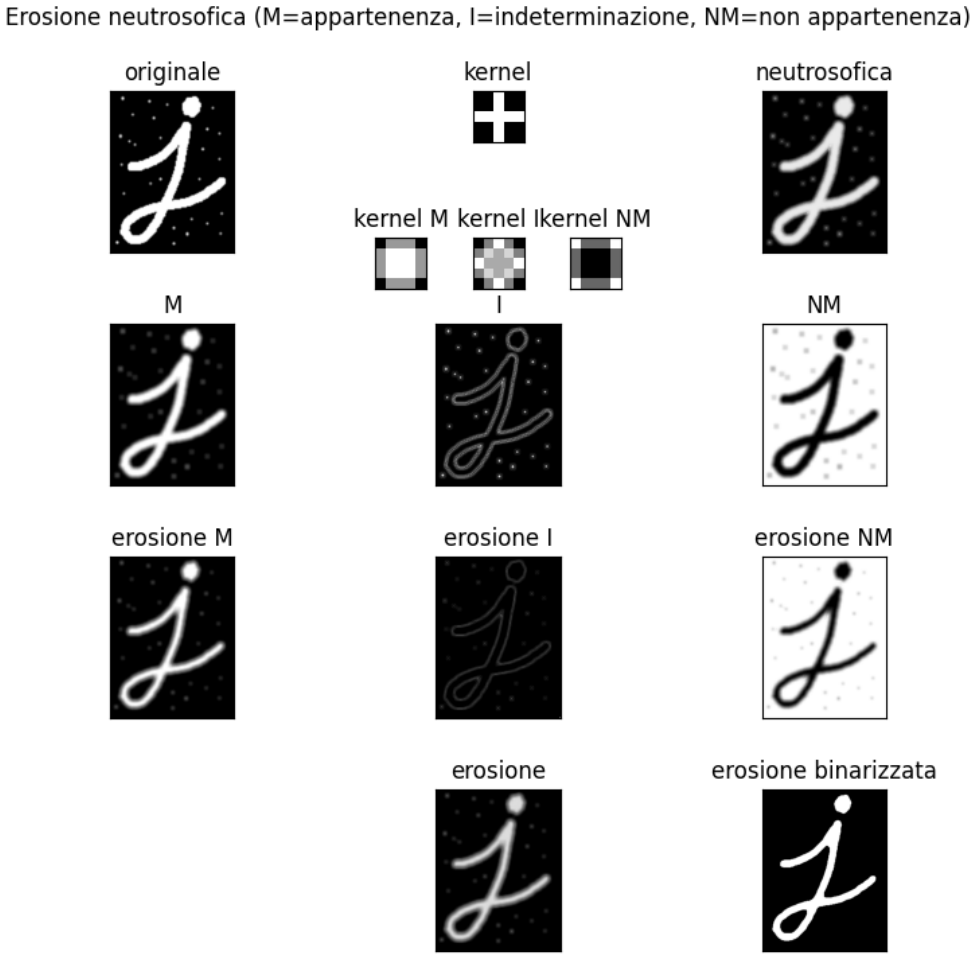


Figura 5.3: Immagine neutrosofica di una j in corsivo con rumore di tipo sale e sua erosione con raggio immagine = 4 e raggio kernel =1

non appartenenza, ma nonostante ciò non altera l'azione tipica dell'erosione, ovvero quella di eliminare eventuali sporgenze dal profilo dell'immagine e di ingrandimento dei fori.

Analogamente, nella Figura 5.4, si può notare che l'operatore di apertura neutrosofica applicato all'immagine col rumore mantiene la traccia del rumore stesso. Esso è maggiormente visibile sia nella componente I di indeterminazione sia nella componente NM di non appartenenza, ma nonostante ciò non altera l'azione tipica dell'apertura, cioè quella di affinare l'immagine in primo piano lasciando inalterati i disturbi che possono in ogni caso essere soppressi dalla successiva binarizzazione.

Infine, nella Figura 5.5, è possibile notare che l'operatore di chiusura neutrosofica applicato all'immagine col rumore mantiene la traccia del rumore stesso. Esso è maggiormente visibile nella componente I di indeterminazione, ma nonostante ciò non altera l'azione tipica della chiusura, cioè quella di smussamento del profilo dell'immagine e dell'eliminazione di eventuali sporgenze.

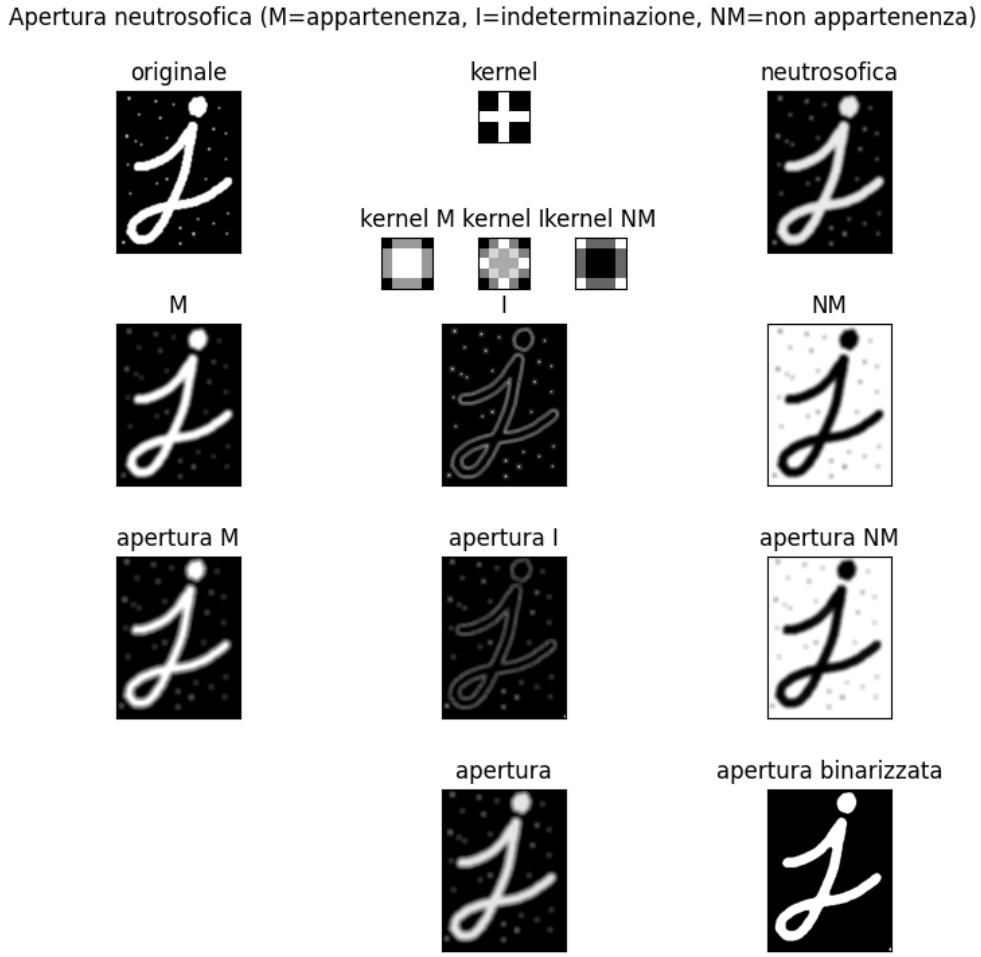


Figura 5.4: Immagine neutrosofica di una j in corsivo con rumore di tipo sale e sua apertura con raggio immagine = 4 e raggio kernel =1

5.5 Comparazioni, Analisi e Conclusioni

In quest'ultima parte, utilizzeremo la classe NSmorph ed alcuni programmi Python che la richiamano per applicare i quattro operatori morfologici di dilatazione, erosione, apertura e chiusura neutrosofica ad alcune immagini binarie provenienti da applicazioni pratiche del mondo reale, ne valuteremo l'efficacia e, dopo averne ottenuto le immagini binarizzate, effettueremo un confronto analitico con le corrispondenti immagini ottenute applicando i rispettivi operatori morfologici binari della Morfologia Matematica classica al fine di stabilire se il nostro approccio neutrosofico alla Morfologia costituisce realmente un miglioramento dal punto di vista operativo.

A tale scopo, considereremo un set di tre immagini binarie provenienti da ambiti piuttosto differenti e precisamente:

- l'immagine ingrandita al microscopio di un gruppo di cellule (Figura 5.6 (a))
- l'immagine di una porzione di un circuito elettronico contenente alcune piste e piazzole di saldatura (Figura 5.11 (a))

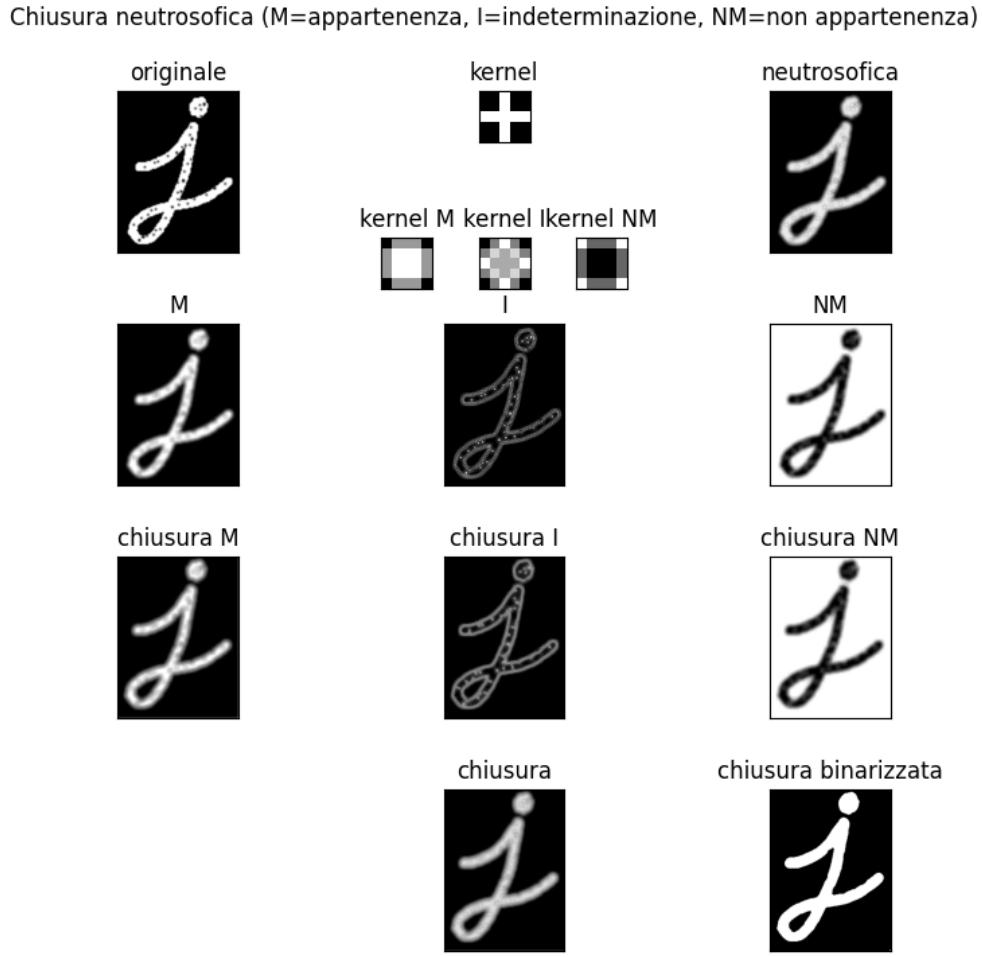


Figura 5.5: Immagine neutrosofica di una j in corsivo con rumore di tipo pepe e sua chiusura con raggio immagine = 4 e raggio kernel =1

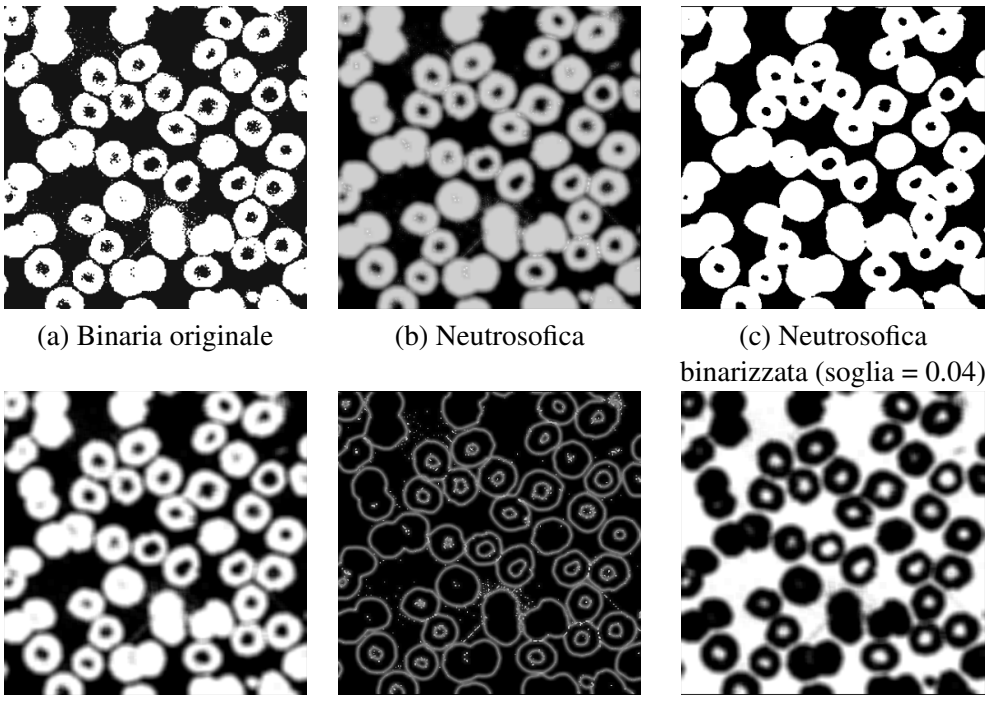
- l'impronta digitale di un polpastrello (Figura 5.16 (a))

e per ciascuna di esse effettueremo le operazioni di dilatazione neutrosofica, erosione neutrosofica, apertura neutrosofica e chiusura neutrosofica, ottenendone le corrispondenti immagini neutrosofiche interpolate e le rispettive componenti di appartenenza (M), indeterminazione (I) e non appartenenza (NM) nonché la sua binarizzazione (ottenuta per sogliatura) al fine di confrontarle, per differenza di pixel, con i risultati ottenuti applicando alle stesse immagini binarie di partenza i corrispondenti operatori di dilatazione, erosione, apertura e chiusura della Morofologia Matematica classica.

Dall'osservazione della Figura 5.6 si può evincere che fin dalla conversione dell'immagine da binaria a neutrosofica, i rumori di tipo sale vengono fortemente attenuati ed i profili delle cellule sono esaltati nella componente di indeterminatezza della stessa (vedasi Figura 5.6 (e)).

Nella Figura 5.7, l'applicazione dell'operatore di dilatazione neutrosofica ha l'effetto di rimuovere i contorni delle singole cellule, sfumandone i bordi, nella componente di appartenenza. D'altra parte, la componente di indeterminazione fa emergere il rumore di tipo sale con una nitidezza a dir poco considerevole.

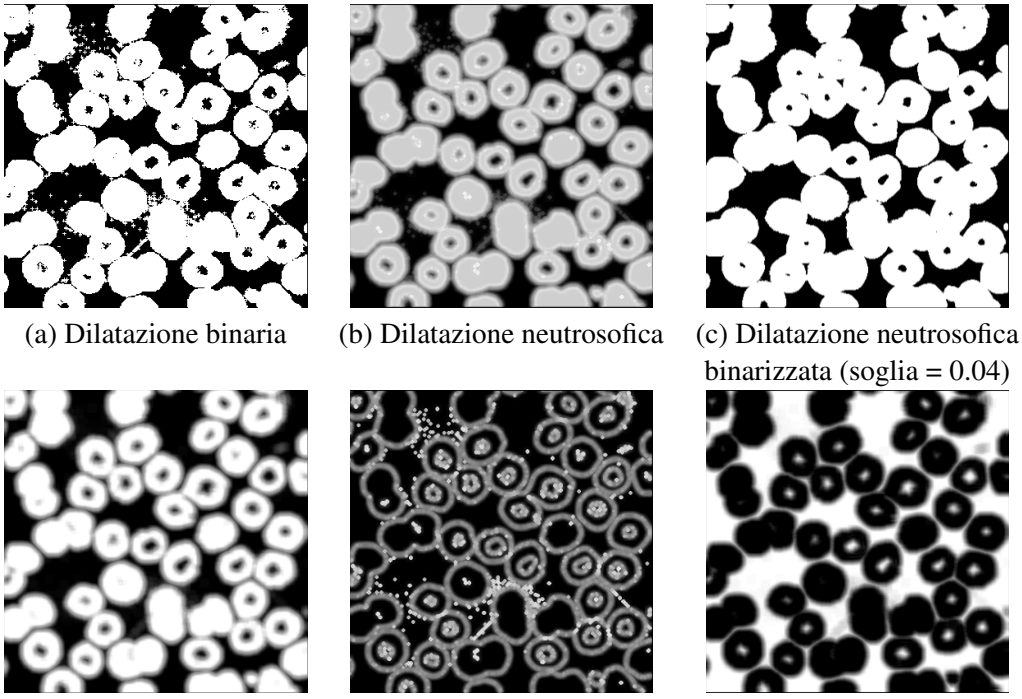
Figura 5.6: Immagine neutrosofica di un gruppo di cellule con raggio $= 4$



(a) Binaria originale (b) Neutrosofica (c) Neutrosofica binarizzata (soglia = 0.04)

(d) Appartenenza (M) (e) indeterminazione (I) (f) Non appartenenza (NM)

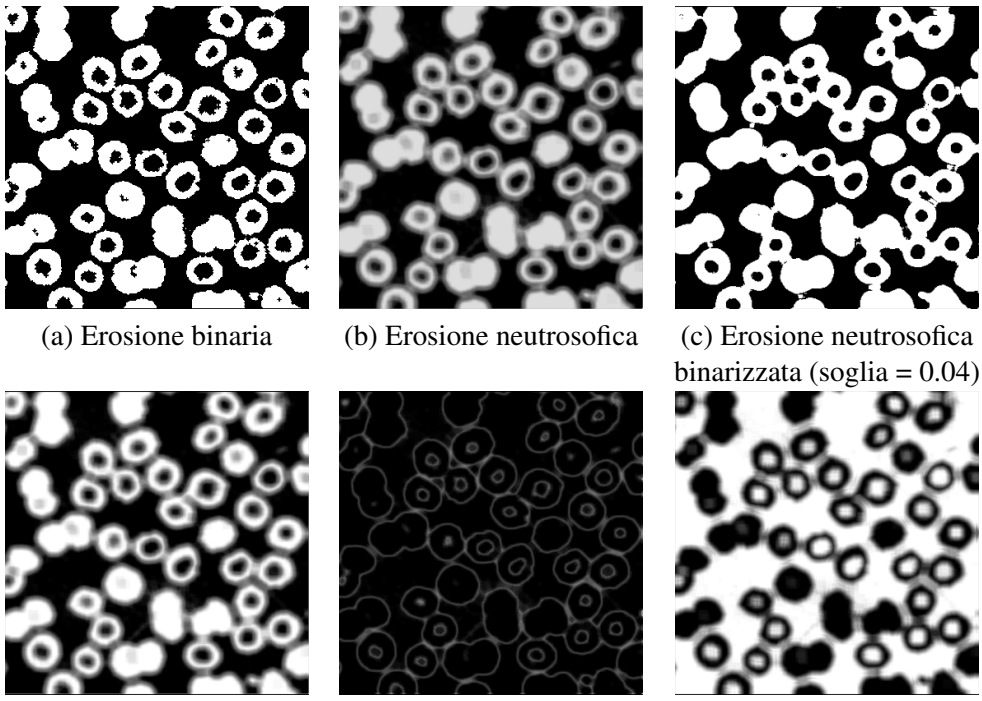
Figura 5.7: Dilatazione neutrosofica di un gruppo di cellule con raggio $= 4$ e raggio kernel croce $= 1$



(a) Dilatazione binaria (b) Dilatazione neutrosofica (c) Dilatazione neutrosofica binarizzata (soglia = 0.04)

(d) Dilatazione M (e) Dilatazione I (f) Dilatazione NM

Figura 5.8: Erosione neutrosofica di un gruppo di cellule con raggio $= 4$ e raggio kernel croce $= 1$



(a) Erosione binaria (b) Erosione neutrosofica (c) Erosione neutrosofica binarizzata (soglia = 0.04)

(d) Erosione M (e) Erosione I (f) Erosione NM

Nella Figura 5.8, l'erosione neutrosofica produce un'immagine neutrosofica ben equilibrata. Se da un lato vi è una chiusura totale dei nuclei di alcune cellule, dall'altro lato la maggior parte dei nuclei delle cellule stesse, non solo è stata preservata, ma ne è stata smussata la forma rendendoli più nitidi.

Nella Figura 5.9, l'apertura neutrosofica ha consentito di rimuovere qualche residuo di rumore di tipo sale dall'immagine di partenza e ne ha migliorato i contorni.

Dall'osservazione della Figura 5.10 si evince che, rispetto all'immagine binaria, alcuni fori più piccoli sono stati chiusi, ma ciò è di gran lunga compensato dal fatto che i contorni delle cellule risultano nettamente più nitidi e definiti.

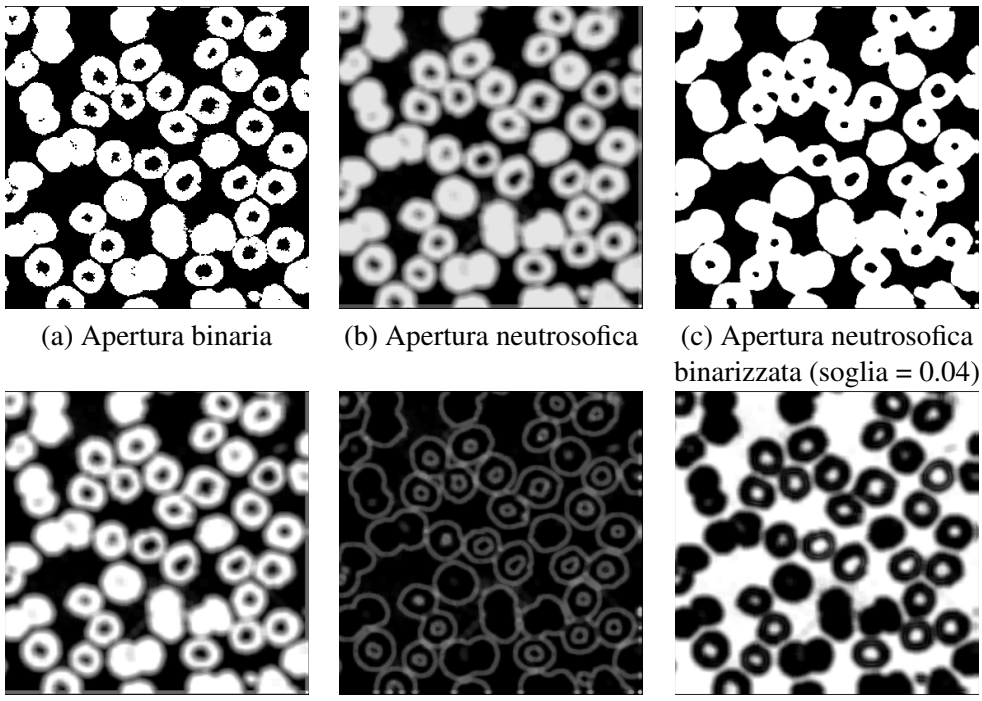
Da una prima osservazione della Figura 5.11 si può osservare che l'immagine neutrosofica del circuito elettrico è molto più definita rispetto a quella originale, soprattutto la pista centrale che appare molto più uniforme. Inoltre i contorni sono perfettamente delineati e messi in risalto nella Figura 5.11 (e) della componente di indeterminazione (I).

Nella Figura 5.12, la dilatazione neutrosofica ha semplificato alcuni punti di saldatura mettendo in maggiore risalto le piste del circuito elettrico uniformandone i bordi.

Tra tutte le immagini osservate fino ad ora, la Figura 5.13 dell'erosione neutrosofica è la più significativa. Infatti nessuna pista, anche la più sottile, è stata cancellata e la forma delle piazzole è stata mantenuta senza cancellare gran parte dei punti di saldatura.

Nella Figura 5.14, l'apertura neutrosofica ha consentito di salvaguardare la visibilità della maggior parte dei punti di saldatura. D'altra parte, ed è qui il lato migliore dell'applicazione di questo operatore, l'apertura binaria (vedasi Figura 5.14 (a)) ha eliminato quasi tutta la pista della parte destra del circuito, mentre l'apertura neutrosofica non solo l'ha preservata, ma l'ha anche estremamente definita nella forma.

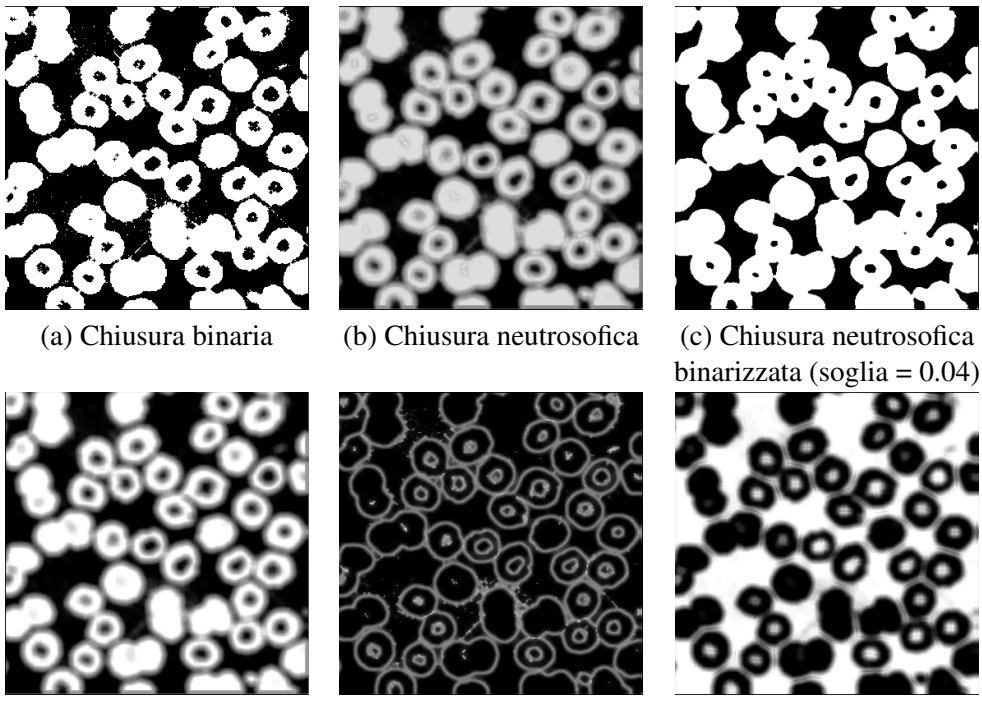
Figura 5.9: Apertura neutrosofica di un gruppo di cellule con raggio $= 4$ e raggio kernel croce $= 1$



(a) Apertura binaria (b) Apertura neutrosofica (c) Apertura neutrosofica binarizzata (soglia = 0.04)

(d) Apertura M (e) Apertura I (f) Apertura NM

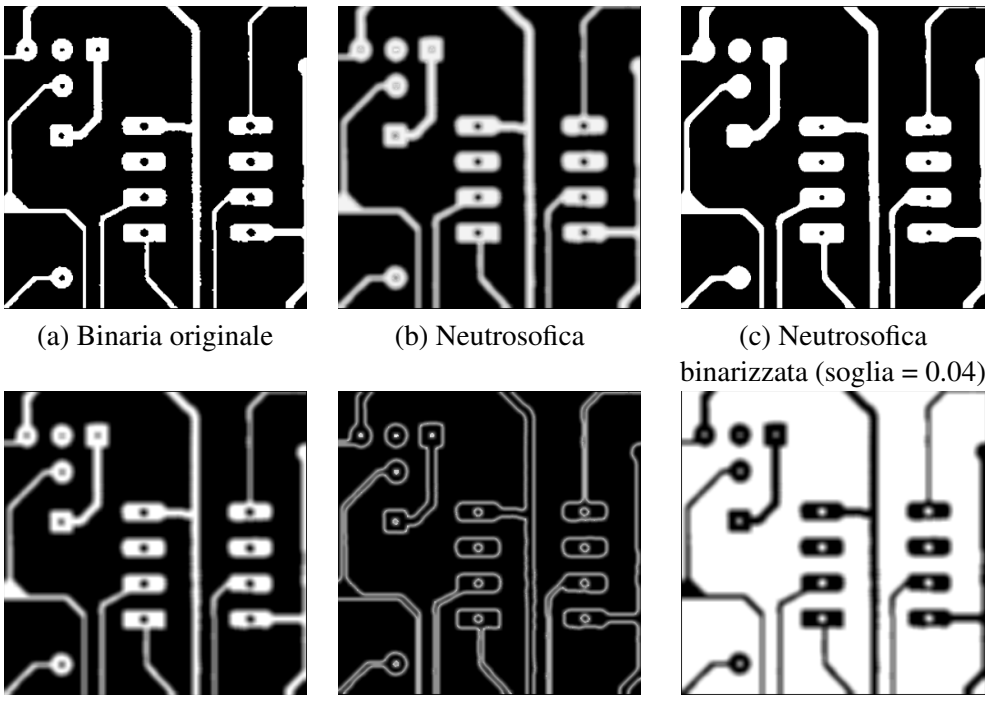
Figura 5.10: Chiusura neutrosofica di un gruppo di cellule con raggio $= 4$ e raggio kernel croce $= 1$



(a) Chiusura binaria (b) Chiusura neutrosofica (c) Chiusura neutrosofica binarizzata (soglia = 0.04)

(d) Chiusura M (e) Chiusura I (f) Chiusura NM

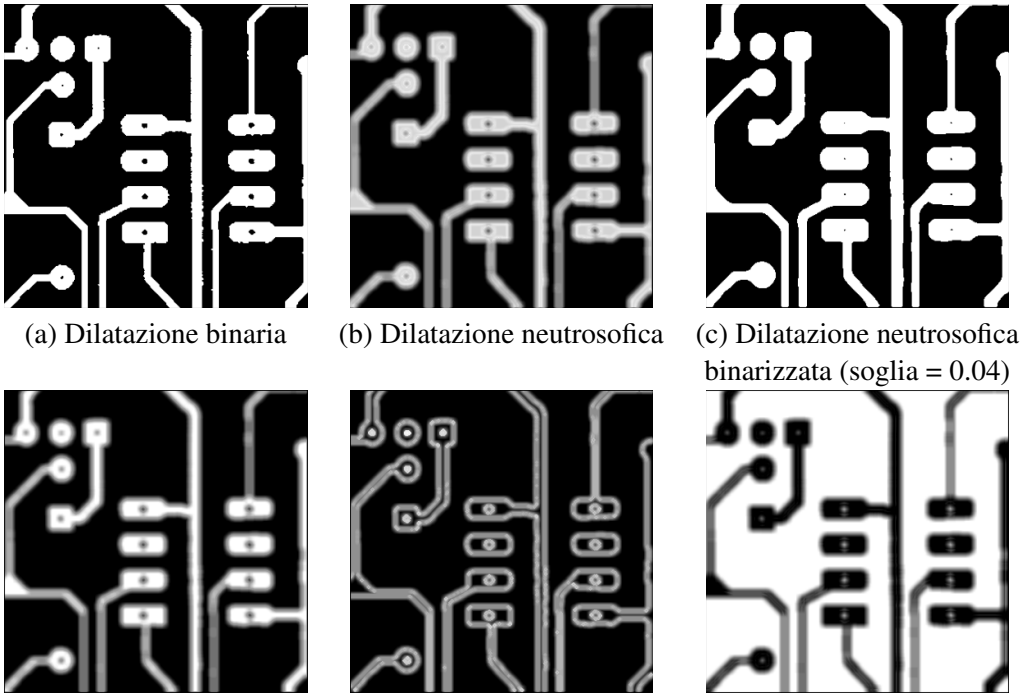
Figura 5.11: Immagine neutrosofica di un circuito elettrico con raggio $= 4$



(a) Binaria originale (b) Neutrosofica (c) Neutrosofica binarizzata (soglia = 0.04)

(d) Appartenenza (M) (e) indeterminazione (I) (f) Non appartenenza (NM)

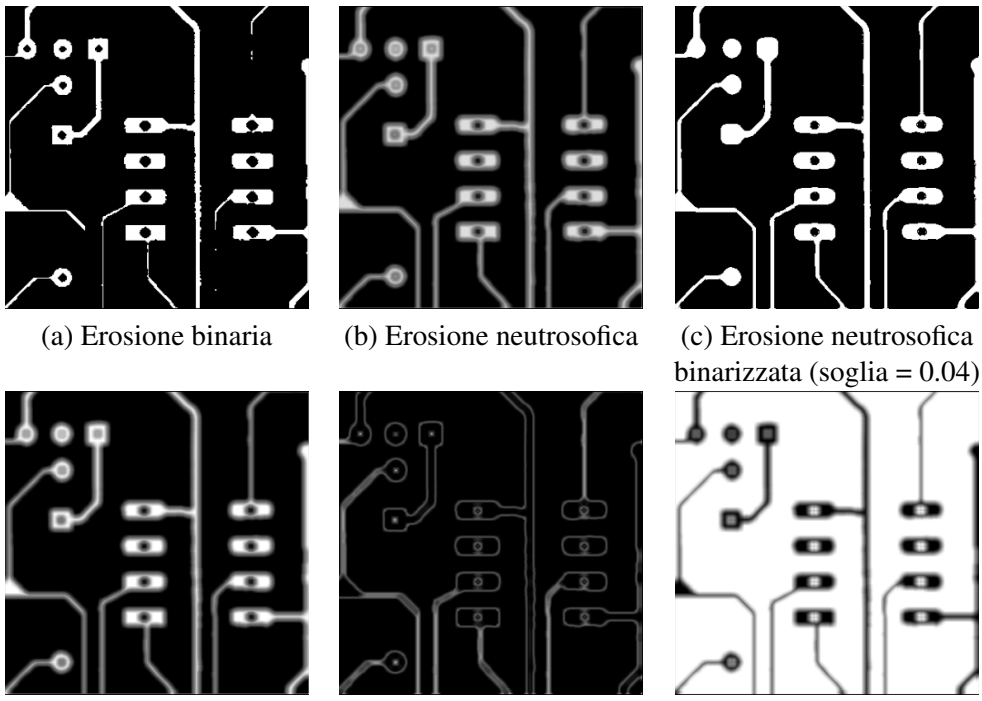
Figura 5.12: Dilatazione neutrosofica di un circuito elettrico con raggio $= 4$ e raggio kernel croce $= 1$



(a) Dilatazione binaria (b) Dilatazione neutrosofica (c) Dilatazione neutrosofica binarizzata (soglia = 0.04)

(d) Dilatazione M (e) Dilatazione I (f) Dilatazione NM

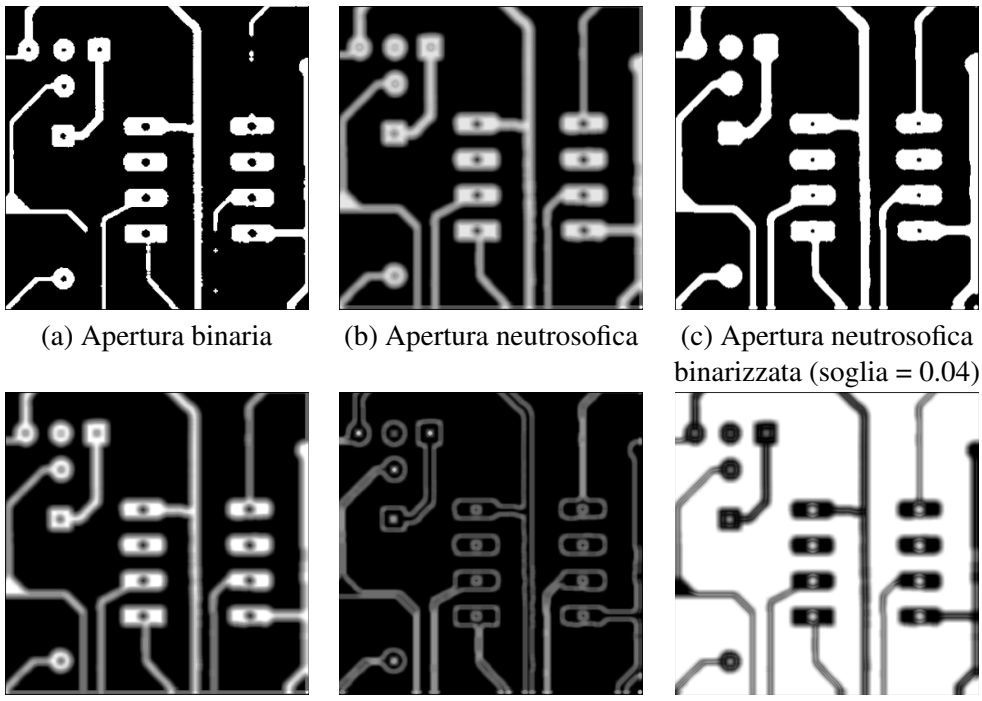
Figura 5.13: Erosione neutrosofica di un circuito elettrico con raggio $= 4$ e raggio kernel croce $= 1$



(a) Erosione binaria (b) Erosione neutrosofica (c) Erosione neutrosofica binarizzata (soglia = 0.04)

(d) Erosione M (e) Erosione I (f) Erosione NM

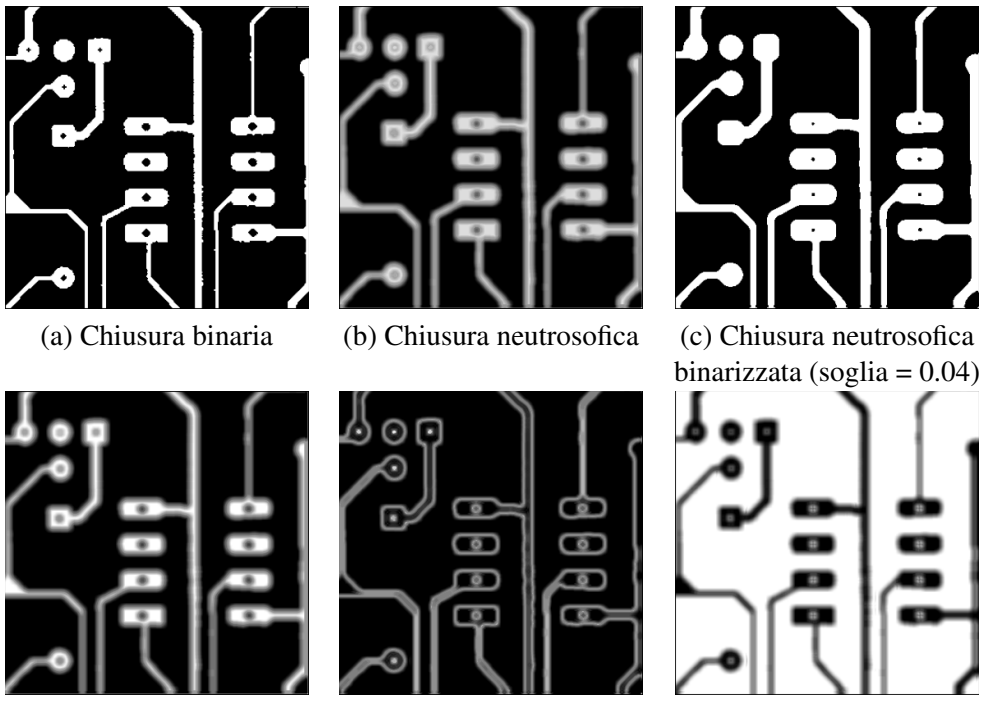
Figura 5.14: Apertura neutrosofica di un circuito elettrico con raggio $= 4$ e raggio kernel croce $= 1$



(a) Apertura binaria (b) Apertura neutrosofica (c) Apertura neutrosofica binarizzata (soglia = 0.04)

(d) Apertura M (e) Apertura I (f) Apertura NM

Figura 5.15: Chiusura neutrosofica di un circuito elettrico con raggio $= 4$ e raggio kernel croce $= 1$



(a) Chiusura binaria (b) Chiusura neutrosofica (c) Chiusura neutrosofica binarizzata (soglia = 0.04)

(d) Chiusura M (e) Chiusura I (f) Chiusura NM

Nella chiusura neutrosofica in Figura 5.15, i punti di saldatura risultano puntiformi e le piste sono ben definite ed uniformi.

Da una prima osservazione della Figura 5.16, si può notare come le criste dell'impronta digitale risultino più evidenti e smussate nell'immagine neutrosofica ed in particolare nella componente di indeterminazione (I) in Figura 5.16 (e).

Nella Figura 5.17, a differenza della dilatazione binaria che cancella quasi del tutto le criste dell'impronta digitale, la dilatazione neutrosofica, invece, ne preserva la continuità delle stesse pur rimuovendo il rumore di fondo costituito dall'effetto sale.

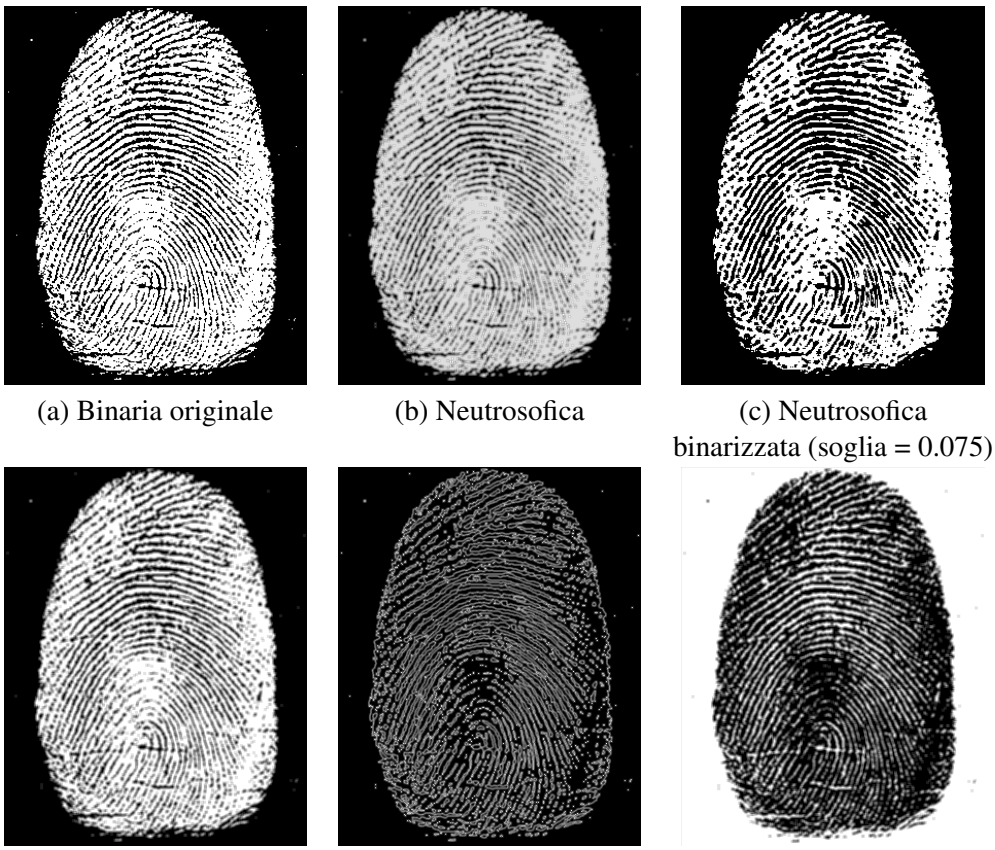
Dall'osservazione della Figura 5.18 è evidente che l'erosione binaria elimina quasi del tutto i solchi dell'impronta digitale, invece l'erosione neutrosofica, nonostante crei alcuni limitatissimi addensamenti, ne preserva la continuità .

Nella Figura 5.19 è interessante osservare che l'apertura binaria crea dei veri e propri solchi in corrispondenza della parte inferiore sinistra e della parte centrale dell'impronta, mentre l'apertura neutrosofica, nonostante alcuni addensamenti, preserva la continuità e la forma delle criste.

Dall'osservazione della Figura 5.20 si evidenzia una sostanziale differenza qualitativa tra la chiusura binaria e la chiusura neutrosofica, in quanto la chiusura binaria (in Figura 5.20 (a)) ha eliminato quasi del tutto le criste dell'impronta digitale lasciando solo alcuni fori sparsi, mentre la chiusura neutrosofica ha accentuato i solchi dell'impronta, ma ne ha preservato la forma e la struttura.

Nel confrontare digitalmente e visivamente le immagini binarie corrispondenti alla dilatazione classica ed alla dilatazione neutrosofica, emerge chiaramente (si veda il profilo rosa della Figura 5.22 (c)) come l'operazione di dilatazione sia stata accompagnata da una contestuale pulizia del rumore di tipo sale.

Figura 5.16: Immagine neutrosofica di un'impronta digitale con raggio $= 1$



(a) Binaria originale (b) Neutrosofica (c) Neutrosofica binarizzata (soglia = 0.075)

(d) Appartenenza (M) (e) indeterminazione (I) (f) Non appartenenza (NM)

Da un primo confronto visivo tra le immagini binarie corrispondenti all'erosione classica ed all'erosione neutrosofica, è evidente che i nuclei più piccoli dell'immagine neutrosofica vengono rimossi, ma la maggior parte di essi sono stati rimpiccioliti e smussati, come si può osservare nella Figura 5.23 (c).

Nel confrontare digitalmente le immagini binarie corrispondenti all'apertura classica ed all'apertura neutrosofica in Figura 5.24, emerge chiaramente come l'operazione di apertura neutrosofica abbia ridefinito uniformemente i contorni delle pareti cellulari e dei nuclei, con un conseguente rimpicciolimento degli stessi e l'eliminazione dei nuclei più piccoli.

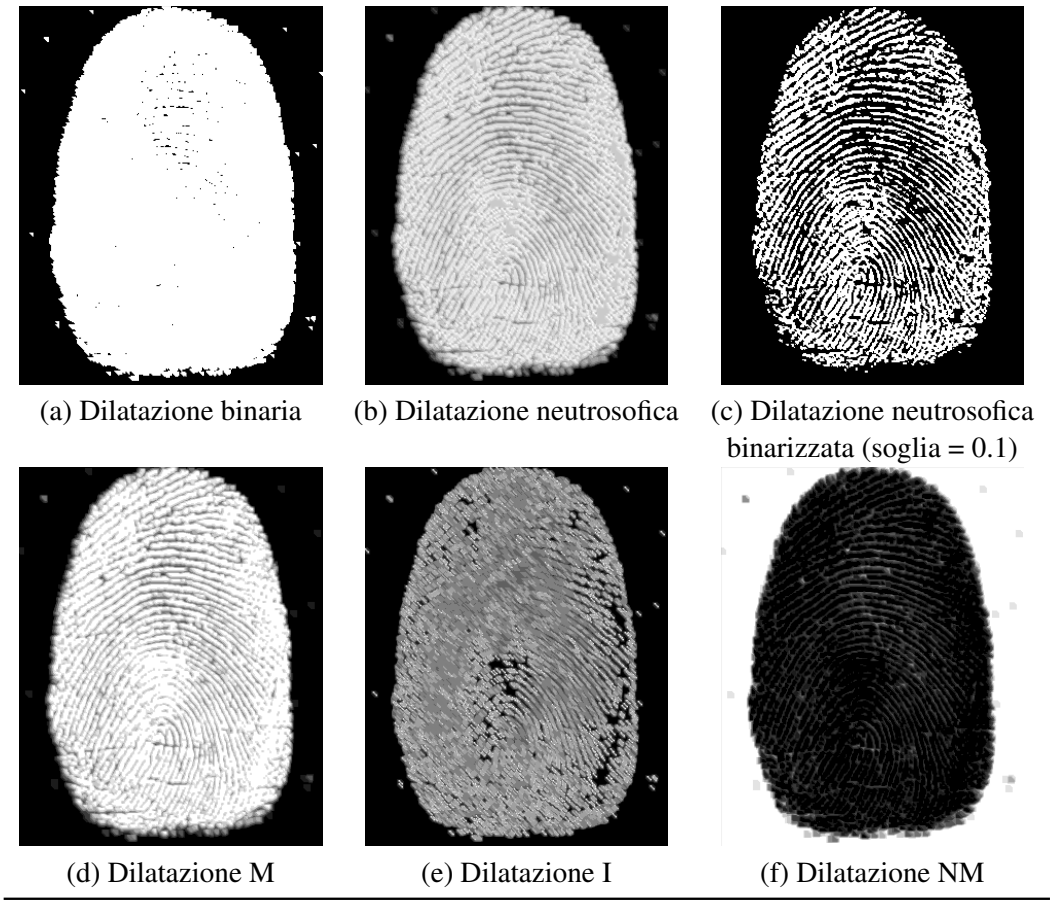
Nel confronto tra la chiusura binaria e la chiusura neutrosofica binarizzata in Figura 5.25, si apprezza, in modo più evidente rispetto al confronto tra le immagini binarie corrispondenti all'apertura, un maggiore rimpicciolimento dei nuclei delle cellule.

Nel confronto digitale tra le immagini binarie corrispondenti alla dilatazione classica e la dilatazione neutrosofica si può evincere che la dilatazione neutrosofica mette in evidenza l'ottima definizione delle piste del circuito ed i punti di saldatura, come si può notare dalla Figura 5.26 (c).

Confrontando digitalmente tra le immagini binarie corrispondenti all'erosione classica e all'erosione neutrosofica si può notare come l'erosione neutrosofica evidenzi l'ottima definizione delle piste del circuito che sono state leggermente smussate e rese più uniformi, contrariamente a quanto accaduto nell'erosione classica in cui alcune piste del circuito sono quasi del tutto eliminate, come si può notare dalla Figura 5.27 (c).

Nelle Figure 5.28 e 5.29, nonostante alcuni punti di saldatura vengano eliminati, si può

Figura 5.17: Dilatazione neutrosofica di un'impronta digitale con raggio $= 1$ e raggio kernel triangolare $= 1$



(a) Dilatazione binaria (b) Dilatazione neutrosofica (c) Dilatazione neutrosofica binarizzata (soglia = 0.1)

(d) Dilatazione M (e) Dilatazione I (f) Dilatazione NM

notare come l'apertura e la chiusura neutrosofica mantengano uniformi e continui i profili delle piste del circuito ed i punti di saldatura presenti a differenza dell'apertura classica in Figura 5.28 (a) e della chiusura classica in Figura 5.29 (a). Ciò è reso particolarmente apprezzabile nelle Figure 5.28 (c) e 5.29 (c).

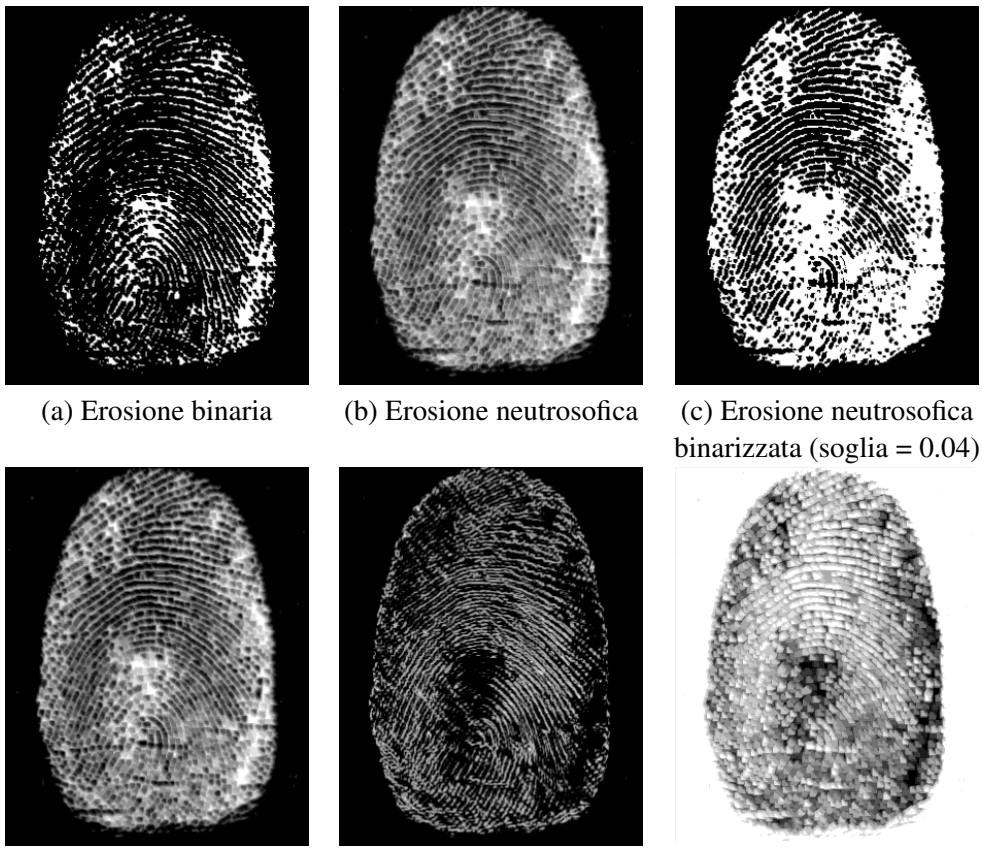
Dal confronto tra le immagini binarie corrispondenti alla dilatazione classica ed alla dilatazione neutrosofica dell'impronta digitale, come già evidenziato nella Figura 5.17, l'operatore di dilatazione neutrosofica preserva la continuità delle criste dell'impronta digitale evidenziate in rosa in Figura 5.30 (c).

Anche dal confronto tra le immagini binarie corrispondenti all'erosione classica ed all'erosione neutrosofica dell'impronta digitale, come già evidenziato nella Figura 5.18, l'operatore di erosione neutrosofica preserva la continuità delle criste dell'impronta digitale e dei solchi evidenziate in Figura 5.31 (c).

Dal confronto tra le immagini binarie corrispondenti all'apertura classica ed all'apertura neutrosofica, si osserva una notevole differenza tra le due immagini evidenziata in rosa in Figura 5.32 (c), poiché l'apertura binaria, pur mantenendo la forma dell'impronta digitale, perde alcune criste sulla parte sinistra rispetto all'apertura neutrosofica che, invece, mantiene un ottimo profilo di impronta e le criste sono sufficientemente visibili.

Infine, dal confronto tra le immagini binarie corrispondenti alla chiusura classica ed alla chiusura neutrosofica, si osserva una notevole differenza tra le due immagini evidenziata in rosa in Figura 5.33 (c), poiché, in questo caso, la chiusura binaria, pur mantenendo la forma

Figura 5.18: Erosione neutrosofica di un'impronta digitale con raggio $= 1$ e raggio kernel triangolare $= 1$



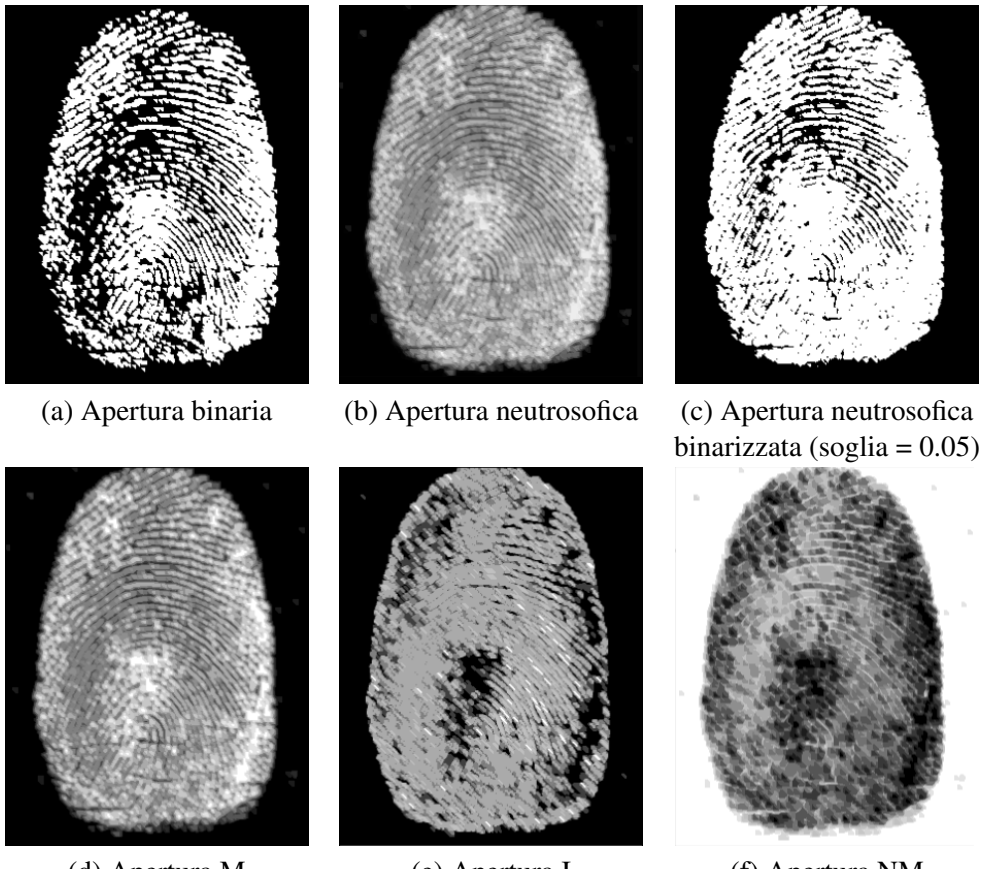
(a) Erosione binaria (b) Erosione neutrosofica (c) Erosione neutrosofica binarizzata (soglia = 0.04)

(d) Erosione M (e) Erosione I (f) Erosione NM

dell'impronta digitale, elimina quasi tutte le criste rispetto all'apertura neutrosofica che, invece, mantiene un buon profilo di impronta e le criste sono molto visibili.

Nel complesso, in tutti e tre gli esempi valutati, si è potuto oggettivamente constatare un aumento della definizione e della qualità delle immagini sottoposte ad operatori morfologici neutrosofici rispetto ai corrispondenti operatori morfologici classici. Ciò avvalora, dunque, la giustezza del nostro approccio.

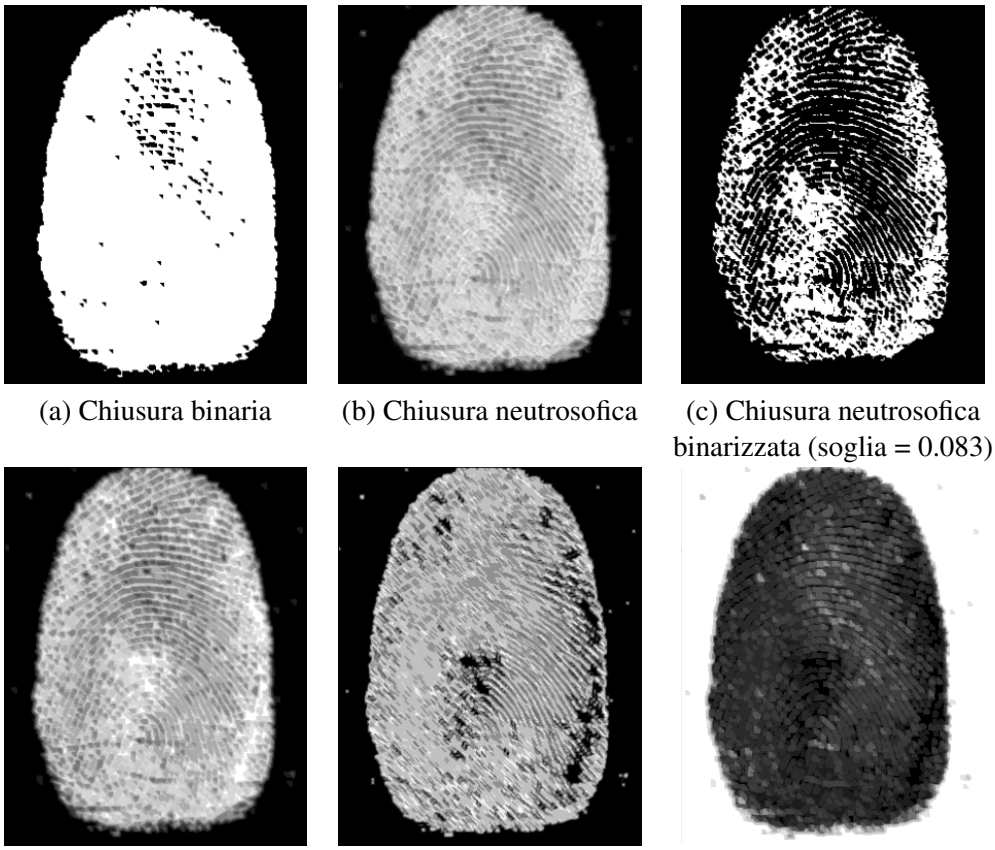
Figura 5.19: Apertura neutrosofica di un'impronta digitale con raggio $= 1$ e raggio kernel triangolare $= 1$



(a) Apertura binaria (b) Apertura neutrosofica (c) Apertura neutrosofica binarizzata (soglia = 0.05)

(d) Apertura M (e) Apertura I (f) Apertura NM

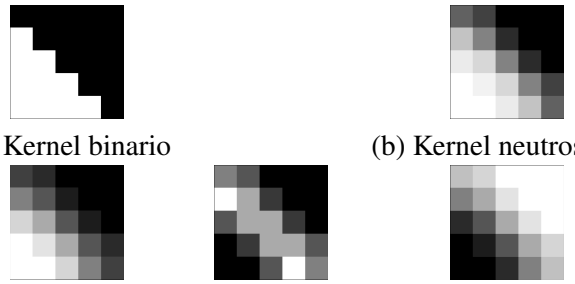
Figura 5.20: Chiusura neutrosofica di un'impronta digitale con raggio $= 1$ e raggio kernel triangolare $= 1$



(a) Chiusura binaria (b) Chiusura neutrosofica (c) Chiusura neutrosofica binarizzata (soglia = 0.083)

(d) Chiusura M (e) Chiusura I (f) Chiusura NM

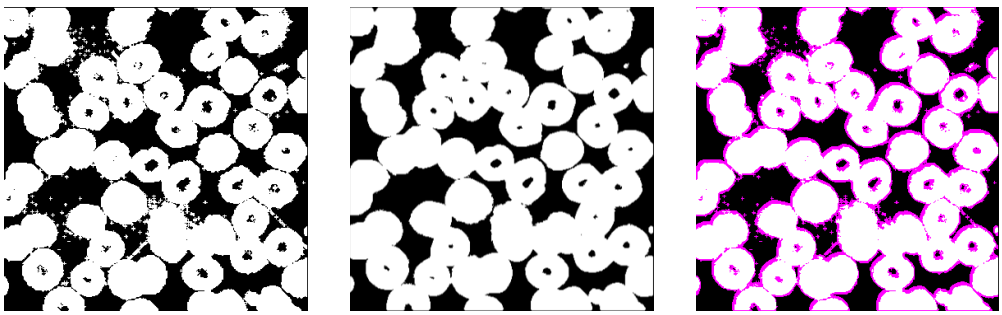
Figura 5.21: Kernel triangolare di dimensione 5×5 e raggio $r = 1$



(a) Kernel binario (b) Kernel neutrosofico

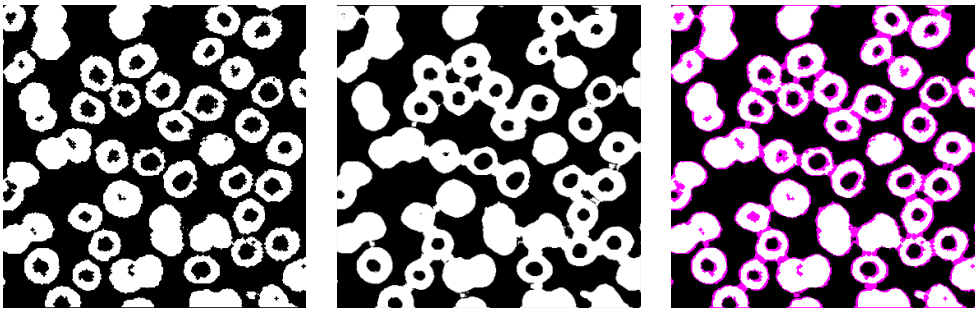
(c) Kernel M (d) Kernel I (e) Kernel NM

Figura 5.22: Confronto tra dilatazione binaria e dilatazione neutrosofica del gruppo di cellule



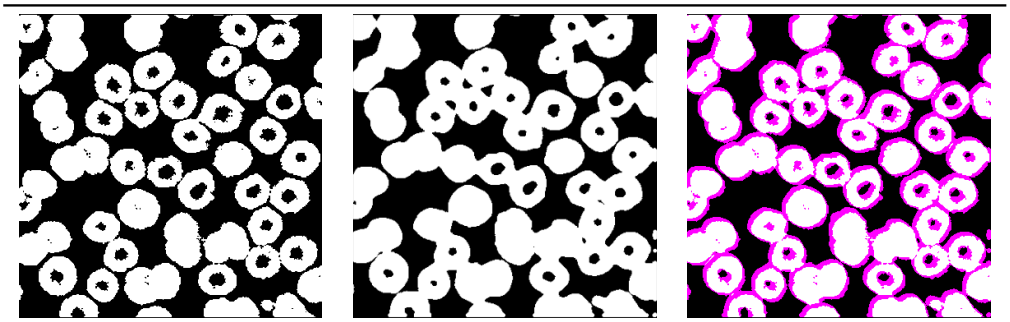
(a) Dilatazione binaria (b) Dilatazione neutrosofica (c) Differenza

Figura 5.23: Confronto tra erosione binaria ed erosione neutrosofica del gruppo di cellule



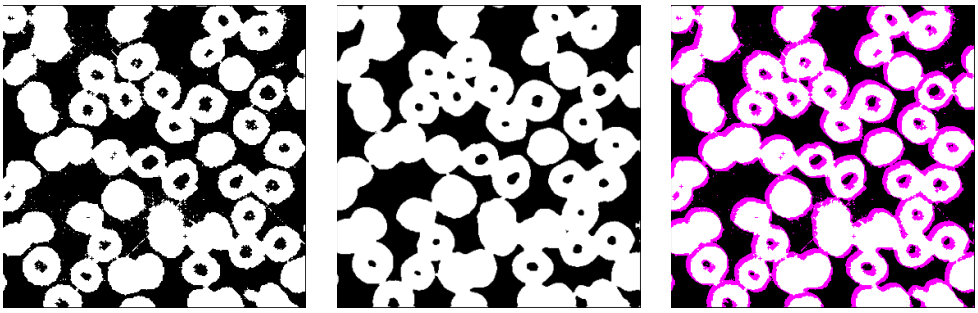
(a) Erosione binaria (b) Erosione neutrosofica (c) Differenza

Figura 5.24: Confronto tra apertura binaria ed apertura neutrosofica del gruppo di cellule



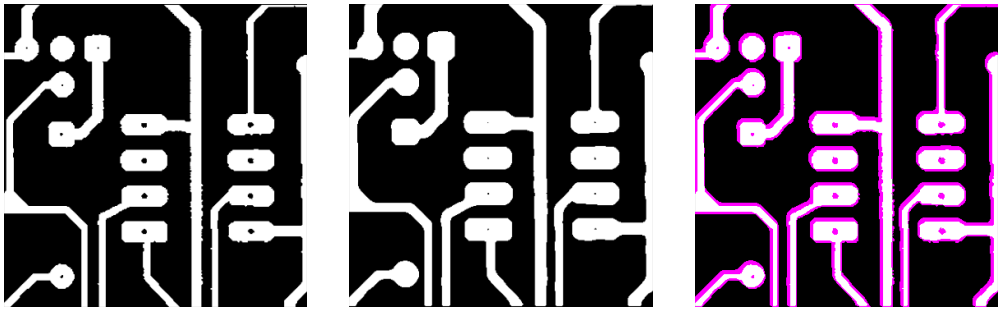
(a) Apertura binaria (b) Apertura neutrosofica (c) Differenza

Figura 5.25: Confronto tra chiusura binaria e chiusura neutrosofica del gruppo di cellule



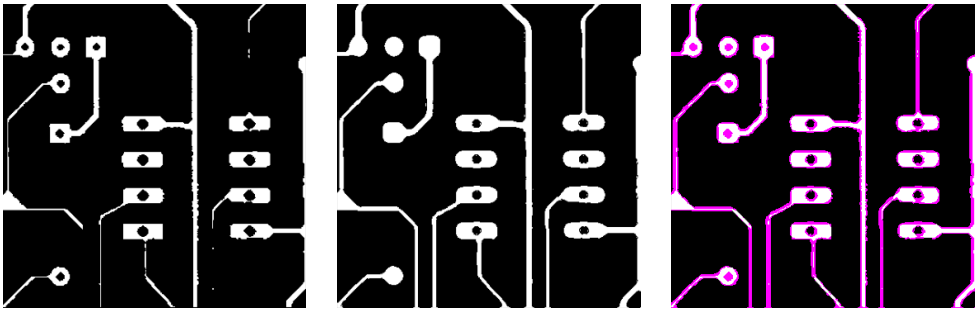
(a) Chiusura binaria (b) Chiusura neutrosofica (c) Differenza

Figura 5.26: Confronto tra dilatazione binaria e dilatazione neutrosofica del circuito



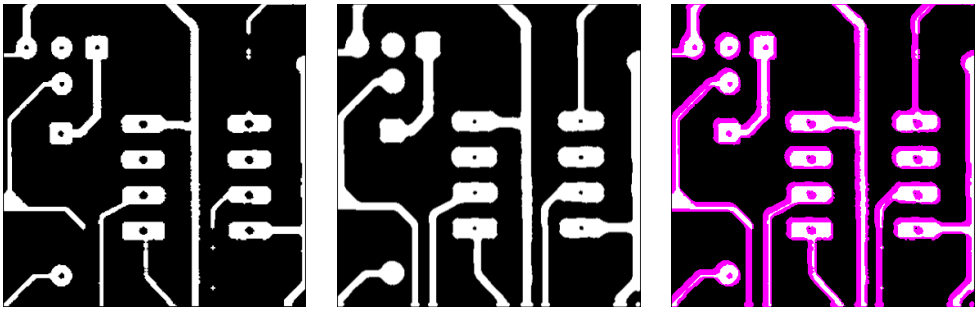
(a) Dilatazione binaria (b) Dilatazione neutrosofica (c) Differenza

Figura 5.27: Confronto tra erosione binaria ed erosione neutrosofica del circuito



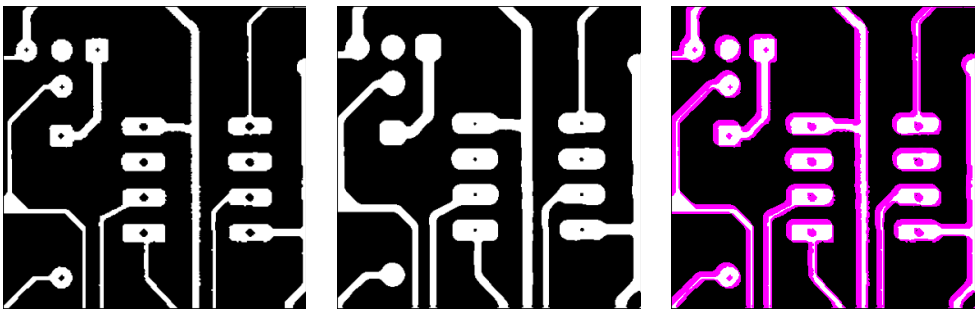
(a) Erosione binaria (b) Erosione neutrosofica (c) Differenza

Figura 5.28: Confronto tra apertura binaria ed apertura neutrosofica del circuito



(a) Apertura binaria (b) Apertura neutrosofica (c) Differenza

Figura 5.29: Confronto tra chiusura binaria e chiusura neutrosofica del circuito



(a) Chiusura binaria (b) Chiusura neutrosofica (c) Differenza

Figura 5.30: Confronto tra dilatazione binaria e dilatazione neutrosofica dell'impronta digitale

(a) Dilatazione binaria (b) Dilatazione neutrosofica (c) Differenza

Figura 5.31: Confronto tra erosione binaria ed erosione neutrosofica dell'impronta digitale

(a) Erosione binaria (b) Erosione neutrosofica (c) Differenza

Figura 5.32: Confronto tra apertura binaria ed apertura neutrosofica dell'impronta digitale

(a) Apertura binaria (b) Apertura neutrosofica (c) Differenza

Figura 5.33: Confronto tra chiusura binaria e chiusura neutrosofica dell'impronta digitale

(a) Chiusura binaria (b) Chiusura neutrosofica (c) Differenza

Appendice

Riportiamo qui di seguito il codice completo della classe Python NSmorph sviluppata in questa tesi.

```
import os.path
import cv2 as cv
import numpy as np
from matplotlib import pyplot as plt

class NSmorph:
    """
    classe che definisce un'immagine neutrosofica a partire da un' immagine in jpg, convertita poi in immagine a toni
    di grigi e che ne fa delle operazioni su di essa, come la dilatazione, l'erosione, l'apertura e la chiusura
    neutrosofica
    """
    #costruttore
    # raggio e' il valore del raggio dell'intorno utilizzato, il valore di default e' 0
    def __init__(self, image, raggio=0):
    """
    costruttore generico di una immagine neutrosofica
    :param image: immagine dalla quale costruire l'immagine neutrosofica, raggio dell'intorno di centro il generico
    pixel (impostato di default a 1)
    """
    #controlli preliminari e sollevamento di eccezioni
    if raggio < 0:
        raise ValueError(f"Il raggio '{raggio}'dell'intorno non puo' essere negativo")
        #Costruzione dell'oggetto rispetto al tipo fornito come parametro
        if type(image) == NSmorph :
            #crea un oggetto copiando un altro oggetto immagine neutrosofica
            #memorizza le proprieta' corrispondenti
            self.__ns_image = image.get()
            self.__image_orig = image.getOrig()
            self.__altezza = image.altezza()
            self.__larghezza = image.larghezza()
            self.__raggio = image.raggio()
        elif type(image) == str:
            #crea un oggetto caricando un file immagine dal disco
            # e ottiene l'immagine neutrosofica utilizzando lo stesso metodo del costruttore
            tmp_imgns = NSmorph(NSmorph.carica(image), raggio)
            #memorizza le proprieta' corrispondenti
            self.__ns_image = tmp_imgns.get()
            self.__image_orig = tmp_imgns.getOrig()
            self.__altezza = tmp_imgns.altezza()
            self.__larghezza = tmp_imgns.larghezza()
            self.__raggio = raggio
```

```
42      elif type(image) == np.ndarray:
43          if image is None:
44              raise ValueError("L'immagine non e' valida")
45          #crea un oggetto partendo da una matrice numpy di valori
46          (altezza, larghezza)=image.shape
47          #memorizziamo le dimensioni dell'immagine
48          self.__altezza = altezza
49          self.__larghezza = larghezza
50          self.__image_orig = image #memorizza come proprieta' l'
    immagine originale fornita
51          self.__raggio = raggio #memorizza come proprieta' il raggio
    dell'intorno
52          #calcola l'immagine neutrosofica
53          #calcolo della funzione intensita' media
54          i_med = np.zeros((altezza, larghezza), dtype = np.float32)
55          for y in range(altezza):
56              for x in range(larghezza):
57                  md = 0
58                  n_pixel = 0
59                  for j in range(y - raggio, y + raggio + 1):
60                      for i in range(x - raggio, x + raggio +1):
61                          if (0<=i<larghezza) and (0<=j<altezza) :
62                              md += image[j][i]
63                              n_pixel += 1
64                  md /= n_pixel
65                  i_med[y][x] = md
66          #Calcoliamo i valori di picco dell'intensita'
67          i_med_min = i_med.min()
68          i_med_max = i_med.max()
69          i_med_ampiezza = i_med_max - i_med_min
70          #Calcoliamo la funzione di omogeneita'
71          delta = np.zeros((altezza, larghezza), dtype=np.float32) #
    matrice di float
72          for y in range(altezza):
73              for x in range(larghezza):
74                  delta[y][x] = abs(image[y][x] - i_med[y][x])
75          #Calcoliamo i valori di picco della funzione omogeneita'
76          delta_min = delta.min()
77          delta_max = delta.max()
78          delta_ampiezza = delta_max - delta_min
79          #Calcolo dei tre livelli vuoti dell'immagine neutrosofica
80          ns_image = np.zeros((altezza, larghezza, 3), dtype = np.
    float32)
81          for y in range(altezza):
82              for x in range(larghezza):
83                  ns_image[y][x][0] = (i_med[y][x] - i_med_min)/
    i_med_ampiezza if i_med_ampiezza !=0 else 0
84                  ns_image[y][x][1] = (delta[y][x] - delta_min)/
    delta_ampiezza if delta_ampiezza !=0 else 0
85                  ns_image[y][x][2] = 1 - ns_image[y][x][0]
86          self.__ns_image = ns_image #memorizza come proprieta' l'
    immagine neutrosofica
87      else:
88          raise ValueError("Il primo parametro deve essere una matrice
     o una stringa")
89
90      #metodo statico che permette il caricamento di una immagine a
    partire dal percorso della stessa
91      @staticmethod
92      def carica(img_path):
93          if img_path is None:
94          raise ValueError("Il percorso dell'immagine non e' valido")
95          #prova a caricare dal disco l'immagine a livelli di grigi
96          if not os.path.isfile(img_path):
97              raise FileNotFoundError(f"Il file '{img_path}' non esiste o
    non e' accessibile.")
```

```
98          image = cv.imread(img_path, cv.IMREAD_GRAYSCALE)
99          if image is None:
100             raise IOError(f"Non e' stato possibile leggere il file
    immagine '{img_path}'")
101         return image
102
103     # metodo che restituisce l'immagine neutrosofica
104     def get(self):
105         """
106               return: immagine neutrosofica
107               """
108         return self.__ns_image
109
110     # metodo che restituisce il grado di appartenenza
111     def getM(self):
112         """
113         return: grado di appartenenza
114         """
115         return self.__ns_image[:,:,0]
116
117     # metodo che restituisce il grado di indeterminatezza
118     def getI(self):
119         """
120         return: grado di indeterminatezza
121         """
122         return self.__ns_image[:,:,1]
123
124     # metodo che restituisce il grado di non appartenenza
125     def getNM(self):
126         """
127         return: grado di non appartenenza
128         """
129         return self.__ns_image[:,:,2]
130
131     # metodo che assegna il livello di appartenenza
132     def setM(self, x, y, mu):
133         """
134         parameters: x ed y sono rispettivamente l'ascissa e l'ordinata
    del pixel (x,y), mu e' il valore del grado di
135         appartenenza da assegnare
136               """
137               self.__ns_image[y][x][0] = mu
138
139     # metodo che assegna il livello di indeterminatezza
140     def setI(self, x, y, sigma):
141         """
142         parameters: x ed y sono rispettivamente l'ascissa e l'ordinata
    del pixel (x,y), sigma e' il valore del grado di
    indeterminatezza da assegnare
143          """
144               self.__ns_image[y][x][1] = sigma
145
146     # metodo che assegna il livello di non appartenenza
147         def setNM(self, x, y, omega):
148         """
149         parameters: x ed y sono rispettivamente l'ascissa e l'ordinata
    del pixel (x,y), omega e' il valore del grado di non
    appartenenza da assegnare
150               """
151               self.__ns_image[y][x][2] = omega
152
153     # metodo che restituisce l'immagine originale
154     def getOrig(self):
155         """
156         return: immagine originale
157         """
```

```
158         return self.__image_orig

160     # metodo che restituisce la larghezza dell'immagine
161     def larghezza(self):
162         """
163         return: larghezza dell'immagine
164         """
165         return self.__larghezza

167     # metodo che restituisce l'altezza dell'immagine
168     def altezza(self):
169         """
170         return: altezza dell'immagine
171         """
172         return self.__altezza

174     #metodo che restituisce il raggio dell'immagine neutrosofica
175     def raggio(self):
176         """
177         return: raggio dell'immagine
178         """
179         return self.__raggio

181     #applica il thresholding all'immagine corrente rispetto ad un
         valore di soglia
182     #restituisce l'immagine binaria come matrice numpy di righe e
         colonne con origine (0,0)
183     #in alto a sinistra ed avente valori 0=nero e 1=bianco.
184     def getBinaria(self, soglia):
185         """
186         parameter: soglia per l'applicazione del thresholding method
187         return: immagine binaria
188         """
189         (ret, bin_image) = cv.threshold(self.__image_orig, soglia, 1,
         cv.THRESH_BINARY)
190         return bin_image

192     # metodo che restituisce un'immagine a toni di grigio
193     def getRappresentazione(self, pesoM=0.85, pesoI=0.25, pesoNM=-0.1,
          binaria=False, soglia=128):
194         """
195         parameter: pesoM=0.85, pesoI=0.25, pesoNM=-0.1, binaria=False,
         soglia=128 di default
196         return: immagine a toni di grigio ottenuta attraverso l'
         interpolazione dei tre livelli con peso di default
197         """
198         img_M = self.getM()
199         img_I = self.getI()
200         img_NM = self.getNM()
201         mat_rap = np.zeros((self.__altezza, self.__larghezza, 3), dtype
         =np.float32)
202         for y in range(self.__altezza):
203             for x in range(self.__larghezza):
204                 mat_rap[y][x][0] = pesoM * img_M[y][x] + pesoI * img_I[y
         ][x] + pesoNM * img_NM[y][x]
205         img_rap = cv.cvtColor(mat_rap, cv.COLOR_BGR2GRAY)
206         if binaria == True:
207             (ret, img_rap) = cv.threshold(img_rap, soglia, 1, cv.
         THRESH_BINARY)
208         return img_rap

210     # metodo che restituisce la dilatazione di un'immagine
         neutrosofica
211     # mediante un elemento strutturante passato come parametro
212     def dilatazione(self, kernel):
213         """
```

```
214         parameter: kernel da applicare per la dilatazione
215         return: immagine neutrosofica dilatata
216         """
217         (altezza, larghezza) = (self.__altezza, self.__larghezza)
218         (altezza_k, larghezza_k) = (kernel.altezza(), kernel.larghezza
    ())
219         # assegno a tre variabili i tre livelli di appartenenza dell'
    immagine originaria e del kernel
220         img_M = self.getM()
221         img_I = self.getI()
222         img_NM = self.getNM()
223         kernel_M = kernel.getM()
224         kernel_I = kernel.getI()
225         kernel_NM = kernel.getNM()
226         # ---------------------------
227         # creiamo una matrice generatrice della stessa dimensione di
    quella di partenza
228         # utilizzando lo stesso costruttore della classe ed una matrice
     temporanea tutta nulla
229         mat_generatrice = np.zeros((altezza, larghezza, 3), dtype=np.
    uint8)
230         im_vuota = cv.cvtColor(mat_generatrice, cv.COLOR_BGR2GRAY)
231         # creiamo a partire dall'immagine vuota precedente l'immagine
    neutrosofica im_dil per la dilatazione
232         im_dil = NSmorph(im_vuota) #NS_morph(im_vuota, 4)
233
234         # determiniamo i valori della dilatazione su ogni singolo pixel
     di coordinate (x,y)
235         for y in range(altezza):
236             for x in range(larghezza):
237                 # estraiamo le matrici di appartenenza, indeterminatezza
    e non appartenenza
238                 # della stessa dimensione del kernel a partire dalla
    posizione x,y
239                 mat_M = img_M[y: y + altezza_k, x: x + larghezza_k]
240                 mat_I = img_I[y: y + altezza_k, x: x + larghezza_k]
241                 mat_NM = img_NM[y: y + altezza_k, x: x + larghezza_k]
242                 # calcoliamo le dimensioni effettive delle matrici (che
    risultano essere tutte uguali)
243                 # nel caso in cui si arrivi vicino ai bordi (destro ed
    inferiore)
244                 (n_righe, n_colonne)=mat_M.shape
245
246                 # riduciamo le componenti del kernel alle dimensioni
    della matrice estratta
247                 mat_kernelM = kernel_M[0:n_righe, 0:n_colonne]
248                 mat_kernelI = kernel_I[0:n_righe, 0:n_colonne]
249                 mat_kernelNM = kernel_NM[0:n_righe, 0:n_colonne]
250
251                  # calcoliamo i minimi (per mat_M e mat_I) o i massimi (
    per mat_NM)
252                  # degli elementi corrispondenti delle matrici estratte e
     dei kernel
253                  # (o di 1 meno il kernel nel caso di mat_NM)
254                  minimi_M = np.minimum(mat_M, mat_kernelM)
255                  minimi_I = np.minimum(mat_I, mat_kernelI)
256                  massimi_NM = np.maximum(mat_NM, 1 - mat_kernelNM)
257                  # calcolo dei valori dei gradi di appartenenza,
    indeterminatezza e non appartenenza
258                  # dei pixel di coordinate (x,y) come sup (cioe' il
    massimo) o l'inf (cioe' il minimo)
259                  # delle matrici ottenute
260                  mu = minimi_M.max()
261                  sigma = minimi_I.max()
262                  omega = massimi_NM.min()
```

```
263             # memorizziamo i valori dei tre gradi di appartenenza
    nell'immagine dilatata
264             # in corrispondenza del pixel di coordinate (x,y)
265             im_dil.setM(x, y, mu)
266             im_dil.setI(x, y, sigma)
267             im_dil.setNM(x, y, omega)
268         return im_dil

270     # metodo che restituisce l'erosione di un'immagine neutrosofica
    mediante un elemento strutturante
271     def erosione(self, kernel):
272         """
273         parameter: kernel da applicare per l'erosione
274         return: immagine neutrosofica erosa
275         """
276         (altezza, larghezza) = (self.__altezza, self.__larghezza)
277         (altezza_k, larghezza_k) = (kernel.altezza(), kernel.larghezza
    ())
278         # assegniamo a tre variabili i tre livelli di appartenenza dell
    'immagine origimnaria
279         img_M = self.getM()
280         img_I = self.getI()
281         img_NM = self.getNM()
282         kernel_M = kernel.getM()
283         kernel_I = kernel.getI()
284         kernel_NM = kernel.getNM()
285         # ---------------------------
286         # creiamo una matrice generatrice della stessa dimensione di
    quella di partenza
287         # utilizzando lo stesso costruttore della classe ed una matrice
     temporanea tutta nulla
288         mat_generatrice = np.zeros((altezza, larghezza, 3), dtype=np.
    uint8)
289         im_vuota = cv.cvtColor(mat_generatrice, cv.COLOR_BGR2GRAY)
290         # creiamo a partire dall'immagine vuota precedente l'immagine
    neutrosofica im_er per l'erosione
291         im_er = NSmorph(im_vuota) #NS_morph(im_vuota, 4)
292
293         # determiniamo i valori dell'erosione su ogni singolo pixel di
    coordinate (x,y)
294         for y in range(altezza):
295             for x in range(larghezza):
296                 # estraiamo le matrici di appartenenza, indeterminatezza
    e non appartenenza
297                 # della stessa dimensione del kernel a partire dalla
    posizione x,y
298                 mat_M = img_M[y: y + altezza_k, x: x + larghezza_k]
299                 mat_I = img_I[y:y + altezza_k, x:x + larghezza_k]
300                 mat_NM = img_NM[y:y + altezza_k, x:x + larghezza_k]
301                 # calcoliamo le dimensioni effettive delle matrici (che
    risultano essere tutte uguali)
302                 # nel caso in cui si arrivi vicino ai bordi (destro ed
    inferiore)
303                 (n_righe, n_colonne) = mat_M.shape
304                 # riduciamo le componenti del kernel alle dimensioni
    della matrice estratta
305                 mat_kernelM = kernel_M[0:n_righe, 0:n_colonne]
306                 mat_kernelI = kernel_I[0:n_righe, 0:n_colonne]
307                 mat_kernelNM = kernel_NM[0:n_righe, 0:n_colonne]
308
309                 # calcoliamo i massimi (per mat_M e mat_I) o i minimi (
    per mat_NM)
310                 # degli elementi corrispondenti delle matrici estratte e
    dei kernel
311                 # (o di 1 meno il kernel nel caso di mat_M e mat_I)
312                 massimi_M = np.maximum(mat_M, 1 - mat_kernelM)
```

```
313             massimi_I = np.maximum(mat_I, 1 - mat_kernelI)
314             minimi_NM = np.minimum(mat_NM, mat_kernelNM)
315             # calcolo dei valori dei gradi di appartenenza,
    indeterminatezza e non appartenenza
316             # dei pixel di coordinate (x,y) come sup (cioe' il
    massimo) o l'inf (cioe' il minimo)
317             # delle matrici ottenute
318             mu = massimi_M.min()
319             sigma = massimi_I.min()
320             omega = minimi_NM.max()
321             # memorizziamo i valori dei tre gradi di appartenenza
    nell'immagine erosa
322             # in corrispondenza del pixel di coordinate (x,y)
323             im_er.setM(x, y, mu)
324             im_er.setI(x, y, sigma)
325             im_er.setNM(x, y, omega)
326         return im_er

328     # metodo che restituisce l'apertura di un'immagine neutrosofica
329     # applicando successivamente l'erosione e la dilatazione
    neutrosofica
330     # rispetto allo stesso elemento strutturante passato come
    parametro
331     def apertura(self, kernel):
332         """
333         parameter: kernel da applicare per l'apertura
334         return: immagine neutrosofica aperta
335         """
336         return self.erosione(kernel).dilatazione(kernel)

338     # metodo che restituisce la chiusura di un'immagine neutrosofica
339     # applicando successivamente la dilatazione e l'erosione
    neutrosofica
340     # rispetto allo stesso elemento strutturante passato come
    parametro
341     def chiusura(self, kernel):
342         """
343         parameter: kernel da applicare per la chiusura
344         return: immagine neutrosofica chiusa
345         """
346         return self.dilatazione(kernel).erosione(kernel)
```

Ringraziamenti

Al termine di questo percorso universitario, non posso che tirare le somme su quanto fatto. Sono molto cresciuto e maturato su diversi ambiti: relazionale aprendomi di più alla vita e con più decisione, scientifico a partire dalle prime materie studiate fino alla stesura di una tesi magistrale che, credeteci o no, sono riuscito a concludere dopo non poche difficoltà. Non posso far altro che ringraziare la vita per la possibilità che mi è stata concessa e perché no -essendo credente- anche Dio per aver permesso tutto questo. Vorrei inoltre dedicare la stesura della tesi ed il conseguimento del titolo a tutti quei ragazzi ed a tutte quelle ragazze che per svariati motivi non hanno portato a termine gli studi a causa di sofferenze, violenze e soprusi. A tutti coloro che se ne sono andati via prima di poter conseguire il sogno della proclamazione: questa laurea, questa agognata laurea, è anche per voi, tutti voi.

Grazie al mio relatore, il prof. Giorgio Nordo, per la pazienza e la disponibilità manifestata durante il periodo di stesura.

Grazie alla mia famiglia tutta, dal primo all'ultimo membro e per come in un modo o nell'altro mi hanno fatto sentire il loro abbraccio nei momenti di sconforto.

Grazie a tutti gli amici della Parrocchia S.Clemente, per l'affetto dimostratomi durante tutto il percorso di studi, dalla triennale alla magistrale. Ho sentito la vostra vicinanza, sempre, anche in quei momenti in cui tutto sembrava dovesse andare male.

Grazie a tutti gli amici di S. Gabriele, perché non mi hanno mai fatto sentire fuori posto nonostante le mie difficoltà a partecipare attivamente agli incontri del gruppo.

Grazie a tutti i miei alunni dell'istituto salesiano "Don Bosco" Ranchibile di Palermo, per l'affetto manifestatomi in questo periodo di scuola e per il bene che mi dimostrano ogni giorno.

Grazie a tutti i colleghi docenti ed agli amici dell'istituto salesiano "Don Bosco" Ranchibile di Palermo, in particolare a Walter, Ciccio, Chiara, Pigi, Aurora, Elena, Alessandro, Floriana, Mirko, Alice e Giuliana. Grazie per avermi fatto sentire parte del vostro gruppo e per il bene che mi avete dimostrato dal primo giorno.

Non posso non menzionare adesso le persone più importanti della mia vita.

Grazie mamma, nonostante il nostro rapporto sia diverso da tutti i "classici" rapporti madre-figlio da film americani, nonostante le molte incomprensioni causate da muri alzati o non rotti, sei e sarai la mamma migliore del mondo. Grazie per la tua vicinanza durante le notti insonni pre-esame e per la consolazione che provavi a darmi, consapevole del fatto

che se non fosse cambiata la mia testa, sicuramente potevi fare ben poco. Grazie perché mi hai permesso di studiare: ringraziamento banale, ma non scontato dato che a molti lo studio non è permesso o non se lo possono permettere. Grazie per tutte le passioni che mi hai trasmesso, dalla musica all'arte, dalle scienze alla letteratura, in particolare quella inglese. Grazie perché sei sempre la stessa nonostante tutti i miei cambiamenti -e ne ho fatti parecchi- e perché non ti sei lasciata "turbare" da ciò che ci siamo detti. Grazie perché nel buio della notte, nonostante io non parlassi spesso di ciò che più mi faceva soffrire, sapevo che comunque tu c'eri. Grazie.

Grazie Riccardo, perché come recita la dedica, il tuo coraggio è stato per me motore del mio coraggio. La tua freschezza nei confronti della vita è stata per me vento impetuoso che ha scosso il mio cuore. Il tuo modo di fare, delicato e deciso è stato per me motivo di sana gelosia. Non sono frasi fatte, ma parole che derivano da un profondo senso di riconoscenza, consapevole del fatto che tu, indirettamente, hai fatto tanto per me e spesso mi hai dato l'esempio.

Grazie nonno Giovanni. Ebbene si, siamo arrivati a questo ringraziamento. Non basterebbero le pagine di un romanzo per i milioni di grazie che dovrei dirti. Proverò ad essere sintetico. Grazie nonno perché ci sei sempre stato, dai momenti di gioco ai momenti più seri ed importanti che la vita ci ha posto davanti. Grazie perché da sempre ti sei impegnato a portare avanti la nostra famiglia -e no, non parlo solo a livello economico-, ma perché con il tuo esempio ed i tuoi insegnamenti hai aiutato la mamma a costruire i suoi progetti, hai aiutato la mamma a realizzare il suo sogno più grande che era ed è la famiglia. Grazie per tutte quelle volte che mi hai preso da scuola; non ti importava ci fosse pioggia, neve, grandine o un sole che spaccava le pietre: te le inventavi tutte per venirmi a prendere e per portarmi a casa dove ci aspettava la nonna che aveva già cucinato per noi. Grazie per tutte le manifestazioni di affetto che in modo diretto o indiretto mi hai manifestato, grazie per tutte quelle volte che mi davi una caramella se mi vedevi triste oppure subito dopo un litigio con la mamma. Grazie per tutte quelle volte che hai dato un senso ad una giornata che senso non aveva insegnandomi a fare gli aerei di carta. Grazie perché sei per me un esempio di amore che mai dimenticherò. Grazie per tutto il resto, quel 'resto' che voglio tenere per me e custodire come perla preziosa. Grazie per tutti i "diccillu a quacchi autru" in risposta ai miei ti voglio bene: sappi che valgono molto di più di quanto tu possa pensare. Grazie.

Grazie nonna Licia, anche se non ci sei più da un po' di tempo ormai, volevo ringraziare anche te in questa tesi. Grazie per il bene che mi hai voluto. So che non potrai leggere fisicamente queste parole che ho scritto per te e so anche che già le conosci tutte, ma se fossi stata in vita sicuramente ti avrebbe fatto piacere ricevere un ringraziamento speciale. Grazie.

Grazie Papà per esserti messo a cercarmi le immagini dei kernel e le immagini più difficili da utilizzare e per le chiamate serali in cui ascoltavi le mie lamentele. Le immagini con le 'j' sono nella tesi. Grazie.

Grazie nonna Tania e nonno Santino, per tutti quei "tanto tu sei bravo e ce la fai" che mi avete detto ogni volta che ci vedevamo. Grazie.

Grazie Roberta per essere la mia migliore amica. Grazie perché se non fosse stato per te e per tutte quelle volte in cui con insistenza mi intimavi di prendere in mano la mia vita, io adesso non sarei qui con la serenità che ho adesso. Grazie per il tuo affetto, per il tuo

volermi bene, per la tua testardaggine ad investire su di me. Grazie per tutte le cose che abbiamo vissuto insieme: i pianti, le gioie, le lauree. Grazie per tutto quello che verrà.

Grazie Veronica, perché la tua presenza gioiosa nella mia vita è stata perla preziosa nei momenti in cui non avevo voglia di ridere né di gioire. Grazie per l'amicizia che abbiamo creato e per come la porteremo avanti da ora in poi. Grazie per tutto ciò che abbiamo condiviso e per quello che ancora condivideremo.

Grazie Gabriele, amico e collega. Ci siamo conosciuti cinque anni fa alla prima lezione di algebra della triennale e mai avremmo pensato di poter stringere questo rapporto di complicità che adesso ci lega. Le nostre vite si sono legate subito dal filo invisibile dell'amicizia. Abbiamo condiviso tutto: gli esami della triennale e della magistrale, la prima proclamazione, la seconda (più o meno), le convivenze per preparare le materie, la convivenza a Palermo ed il lavoro da professori sempre nel capoluogo. Non posso non ringraziarti per tutte quelle volte in cui mi sei stato vicino, per tutte quelle volte che mi hai sostenuto nei momenti critici e per tutte quelle volte in cui con la "gentilezza" che caratterizza (qui rido) mi hai consigliato cosa fare. Grazie per tutto quello che abbiamo vissuto insieme, la zip-line a Roma, i viaggi all'estero fatti e per quelli che faremo. Tante volte ci siamo domandati che cosa avremmo fatto e dove saremmo stati se non ci fossimo conosciuti; sono domande alle quali non possiamo rispondere, ma possiamo dire che questa amicizia ci ha portato in alto, ci ha fatto scalare le montagne e ci ha fatto raggiungere vette altissime. Non sappiamo cosa ci riserva il futuro, non sappiamo se rimarremo fisicamente ancora insieme o, inevitabilmente, le nostre vite si allontaneranno. Quello che è certo è che le nostre gioie le condivideremo sempre. Grazie.

Grazie Marco perché nonostante le innumerevoli ed infinite incomprensioni che ci sono e che ci saranno, una cosa posso dire di te che è sicura e che scriverei sui muri perché certa: il tuo volermi bene. Grazie per tutte le volte in cui in maniera esuberante me lo hai dimostrato. Grazie per tutte le avventure che abbiamo vissuto e che ancora vivremo. Grazie per i momenti di ilarità allucinanti che abbiamo vissuto a Vienna, per i 'bye bye' detti agli italiani che vengono dallo 'Stato' di Roma e per 'no la mia carta è smagnetizzata' alla povera commessa austriaca. Grazie.

Grazie Alessandra ed Alessandro, nonostante non viviamo nella stessa città, non posso fare a meno di ringraziarvi per la vostra presenza nella mia vita, per quelle volte in cui mi sono confidato con voi e mai mi avete fatto mancare una parola di conforto e di vicinanza. Siete diventati parte importante della mia vita e tutto quello che verrà vorrò sempre continuare a condividerlo con voi.

Grazie Giacomo e Pippo, perché anche voi come i 'Zazzoni's' avete reso il mio percorso meno ripido facendomi distrarre, consolandomi nel momento di difficoltà con la vostra freschezza e con i vostri modi di fare.

Grazie Aurora, Chiara B., Chiara S., Paola, Elisa, Andrea perché mi avete da sempre fatto sentire a casa. Non posso dimenticarmi la bellissima accoglienza che la prima volta a S. Gabriele mi avete dedicato; né posso dimenticarmi quella volta in cui mi avete organizzato la festa a sorpresa per il mio "addio" che, fortunatamente, non è mai avvenuto. Grazie perché nonostante non ci vediamo spesso, ogni volta che ci riuniamo sembra di conoscersi da sempre.

Un semplice grazie, che racchiude tutto ciò che sono stato, sono e sarò, va a me. Grazie Lo. Grazie.

Bibliografia

[1] Atanassov K.T. Intuitionistic Fuzzy Sets. *Fuzzy Sets and Systems* **20**(1), pp. 87-96, 1986.

[2] Cheng H.D., Guo, Y., Zhang Y., Zhao W. A new neutrosophic approach to image thresholding. *New Mathematics and Natural Computation* **4**(3), pp.291-308, 2008.

[3] Dougherty E.R., Lotufo R.A. Hands-on Morphological Image Processing. *SPIE Press*, Bellingham, Washington (USA), 2003.

[4] Gonzalez R.C., Woods R.E. Digital Image Processing (3rd edition). *Prentice-Hall Inc.*, Upper Saddle NJ (USA), 2006.

[5] Guo Y., Cheng H.D. New Neutrosophic approach to Image Segmentation. *Pattern Recognition* **42**, pp. 587-595, 2009.

[6] Guo Y., Cheng H.D., Zhang Y., Zhao W. A new Neutrosophic Approach to Image Denoising. *"New Mathematics and Natural Computation*, **5**(3), pp. 653-662, 2009.

[7] Latreche A., Barkat O., Milles S., Ismail F. Single valued neutrosophic mappings defined by single valued neutrosophic relations with applications. *Neutrosophic Sets and Systems* **1**, pp. 203-220, 2020.

[8] Nordo G., Mehmood A., Broumi S. Single Valued Neutrosphic Filters. *International Journal of Neutrosophic Science* **1**(1), pp. 1-14, 2020.

[9] Salama A.A., Alagamy H. Neutrosophic Filters. *International Journal of Computer Science Engineering and Information Technologt Research (IJCSEITR)* **3**(1), pp. 307-312, 2013.

[10] Salama A.A., Smarandache F., Kromov V. Neutrosophic Closed Set and Neutrosophic Continuous Functions, *Neutrosophic Sets and Systems* **4**, pp- 4-8, 2014.

[11] Salama A.A., El Ghawalby H., El-Hafeez S.A., El-Nakeeb Eman M. Foundation for Neutrosophic Mathematical Morphology. *New Trends in Neutrosophic Theories and Applications*, 2016.

[12] Salama A.A., El-Hafeez S.A., El-Nakeeb A., El-Hassanein Eman M. Neutrosophic Approach for Mathematical Morphology. *Master's thesis*, Port Said University, 2018.

[13] Serra J. Image Analysis and Mathematical Morphology. *Academic Press Inc.*, London, 1982.

[14] Shih F.Y. Image processing and Mathematical Morphology: Fundamentals and Applications. *CRC Press*, New York (USA), 2009.

[15] Shimaa F.A., El Ghawalby H., Salama A.A. From Image to Neutrosophic Image. *Neutrosophic Sets and Systems*, pp. 1-13, 2015.

[16] Smarandache F. A Unifying Field in Logics. Neutrosophy: Neutrosophic Probability, Set and Logic. Rehoboth: American Research Press, 1999.

[17] Wang H., Smarandache F., Zhang Y.Q., Sunderraman R. Single Valued Neutrosophic Sets. *Technical Sciences and Applied Mathematics*, pp. 10-14, 2012.

[18] Wilson J.N, Ritter G.X. Handbook of Computer Vision Algorithms in Image Algebra (2nd edition). *CRC Press*, New York (USA), 2000.

[19] Zadeh L.A. Fuzzy Sets, *Information and Control* **8**(3), pp. 338-353, 1965.

[20] Zhang M. Novel Approaches to Image Segmentation based on Neutrosophic Logic. *Doctor of Philosophy*, Utah State University, 2010.